NONHALTING ABELIAN NETWORKS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Swee Hong Chan

May 2019

NONHALTING ABELIAN NETWORKS

Swee Hong Chan, Ph.D.

Cornell University 2019

An abelian network is a collection of communicating automata whose state transitions and message passing each satisfy a local commutativity condition. The foundational theory for abelian networks was laid down in a series of papers by Bond and Levine (2016), which mainly focused on networks that halt on all inputs. In this dissertation, we extend the theory of abelian networks to nonhalting networks (i.e., networks that can run forever). A nonhalting abelian network can be realized as a discrete dynamical system in many different ways, depending on the update order. We show that certain features of the dynamics, such as minimal period length, have intrinsic definitions that do not require specifying an update order.

We give an intrinsic definition of the *torsion group* of a finite irreducible (halting or nonhalting) abelian network, and show that it coincides with the critical group of Bond and Levine (2016) if the network is halting. We show that the torsion group acts freely on the set of invertible recurrent components of the trajectory digraph, and identify when this action is transitive.

This perspective leads to new results even in the classical case of sinkless rotor networks (deterministic analogues of random walks). In Holroyd et. al (2008) it was shown that the recurrent configurations of a sinkless rotor network with just one chip are precisely the unicycles (spanning subgraphs with a unique oriented cycle, with the chip on the cycle). We generalize this result to abelian mobile agent

networks with any number of chips. We give formulas for generating series such as

$$\sum_{n\geq 1} r_n z^n = \det(\frac{1}{1-z}D - A)$$

where r_n is the number of recurrent chip-and-rotor configurations with n chips; D is the diagonal matrix of outdegrees, and A is the adjacency matrix. A consequence is that the sequence $(r_n)_{n\geq 1}$ completely determines the spectrum of the simple random walk on the network.

BIOGRAPHICAL SKETCH

Swee Hong Chan was born in Medan, Indonesia. After completing his high school education in Saint Thomas High School in 2008, Swee Hong went to Singapore for his undergraduate study and received a Bachelor of Science with a major in mathematical sciences from Nanyang Technological University in 2012. He then became a research assistant from 2012 to 2014 in Nanyang Technological University. In 2014, he entered the mathematics graduate program at Cornell University in USA.

This dissertation is dedicated to my father Tong Tin Chan, my mother Sui Lien Peo, and my sister Nancy Chan.

I could not have finished this dissertation without the help and encouragement from my advisor, Prof. Lionel Levine. I would also like to thank my committee members Prof. Marcelo Aguiar and Prof. Karola Mészáros for their feedback and support.

ACKNOWLEDGEMENTS

I would like to extend a special thanks to Darij Grinberg for helpful comments on an earlier draft. I would also like to thank my friends Shu Heng Khor and Yuwen Wang for their effort in encouraging me to finish this dissertation. Parts of this dissertation has previously been published in [23] and [24]. Parts of the research that resulted in this dissertation was done under the support of NSF grant DMS-1455272.

TABLE OF CONTENTS

Biographical Sketch . . . iii
Dedication . . . iv
Acknowledgements . . . v
Table of Contents . . . vi

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Atemporal dynamics . . . 2
1.3 Relating atemporal dynamics to traditional dynamics . . . 4
1.4 Computational questions . . . 6
1.5 The torsion group of a nonhalting abelian network . . . 7
1.6 Critical networks . . . 8
1.7 Sandpile networks . . . 9
1.8 Rotor networks and abelian mobile agents . . . 11
1.9 Removal lemma . . . 14
1.10 Summary of notation . . . 14

2 Commutative Monoid **18**
2.1 Injective actions and Grothendieck group . . . 18
2.2 Finite commutative monoid . . . 22

3 Abelian Networks **26**
3.1 Abelian Networks . . . 26
3.2 Legal and complete executions . . . 29
3.3 Locally recurrent states . . . 30
3.4 The production matrix . . . 32
3.5 Subcritical, critical, and supercritical . . . 34
3.6 Examples . . . 37

4 Torsion Group **52**
4.1 The removal lemma . . . 52
4.2 Recurrent components . . . 55
4.3 Construction of the torsion group . . . 67
4.4 Halting case . . . 74

5 Critical Networks: Recurrence **78**
5.1 Recurrent configurations and the burning test . . . 78
5.2 Thief networks . . . 85
5.3 Capacity and level . . . 92
5.4 Stoppable levels . . . 98

6 Critical Networks: Dynamics **110**
6.1 Activity as a component invariant 110
6.2 Near uniqueness of legal executions 119

7 Sandpile networks **124**
7.1 Counting recurrent states 124
7.2 Counting recurrent components 135

8 Rotor and Agent Networks **140**
8.1 Cycle test 143
8.2 Counting recurrent components 149
8.3 Determinantal formula 152

A Classical matrix tree theorem **157**

Bibliography **159**

CHAPTER 1

INTRODUCTION

1.1 Motivation

An *abelian network* is a collection of communicating automata that live at the vertices of a graph and communicate via the edges, for which *changing the order* of certain interactions *has no effect* on the final outcome. (The full definition is spelled out in §3.1.) Deepak Dhar, the inventor of this model, stated the following in regards to the motivation of this definition:

> "In many applications, especially in computer science, one considers such networks where the speed of the individual processors is unknown, and where the final state and outputs generated should not depend on these speeds. Then it is essential to construct protocols for processing such that the final result does not depend on the order at which messages arrive at a processor." [31]

Examples of abelian networks have been studied by different authors through the following complementary viewpoints:

- From a statistical physics point of view, abelian networks (specifically sandpile networks) are used as examples of models that exhibit self-organized criticality [4, 30].
- From a probabilistic point of view, abelian networks (specifically rotor networks) are used as a way to derandomize models such as simple random walks and internal diffusion-limited aggregation [62, 63]

- From an algebraic point of view, abelian networks associate an invariant, in the form of an abelian group or commutative algebra, to its underlying digraph [2].

The foundational theory for abelian networks was laid down in a series of paper by Bond and Levine [15, 16, 17]. To set the stage we recall a few highlights of their results, which apply to abelian networks that terminate in a finite time (also called *halting networks*).

It is proved in [15] that the output and the final state of a halting network depend only on the input and the initial state (and not on the order in which the automata process their inputs). In [16] the halting networks are characterized as those whose production matrix has Perron-Frobenius eigenvalue $\lambda < 1$. In [17] the behavior of a halting network on sufficiently large inputs is expressed in terms of a free and transitive action of the finite abelian group

$$\mathcal{G} := \mathbb{Z}^A/(I - P)K, \tag{1.1}$$

where A is the total alphabet, I is the $A \times A$ identity matrix, P is the production matrix, and K is the total kernel of the network (all defined in Chapter 3). This group generalizes the sandpile group of a finite graph [55, 30, 8].

1.2 Atemporal dynamics

The protagonists of this dissertation are the *nonhalting* abelian networks, which come in two flavors: *critical* ($\lambda = 1$) and *supercritical* ($\lambda > 1$). In either case, there is some input that will cause the network to run forever without halting.

Curiously, the quotient group (1.1) is still well-defined for such a network. In what sense does this group describe the behavior of the abelian network?

To make this question more precise, we should say what we mean by "behavior" of a nonhalting abelian network. A usual approach would fix an update rule, such as one of the following.

- Parallel update: All automata update simultaneously at each discrete time step.
- Sequential update: The automata update one by one in a fixed periodic order.
- Asynchronous update: Each automaton updates at the arrival times of its own independent Poisson process.

Instead, here we take the view that an abelian network is a discrete dynamical system *without a choice of time parametrization*: The trajectory of the system is not a single path but an infinite directed graph encompassing all possible time parameterizations. An update rule assigns to each starting configuration a directed path in this *trajectory digraph*. The study of the digraph as a whole might be called *atemporal dynamics*: dynamics without time. An example of a theorem of atemporal dynamics is Theorem 1.1, which identifies a set of weak connected components of the trajectory digraph on which the torsion subgroup of $\mathcal{G}$ acts freely.

When time is unspecified, what remains of dynamics? Some of the most fundamental dynamical questions are atemporal: Does this computation halt? Is this configuration reachable from that one? Are there periodic trajectories, and of what lengths?

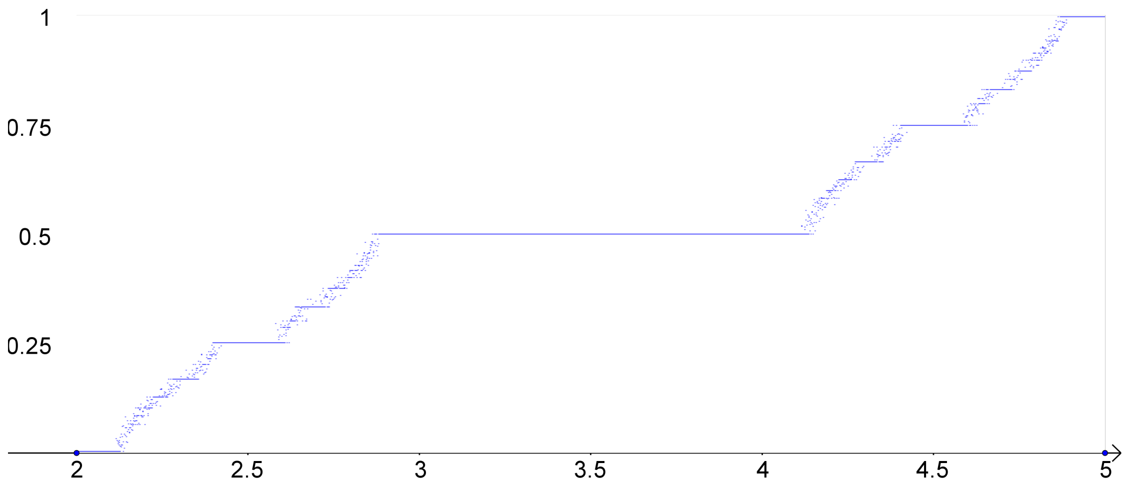

Figure 1.1: A plot of the firing rate (see Definition 6.2) of parallel chip-firing on the discrete torus $\mathbb{Z}_n \times \mathbb{Z}_n$ for $n = 32$. Each point (x, y) represents a random chip configuration with xn^2 chips placed independently with the uniform distribution on the n^2 vertices, and firing rate y.

1.3 Relating atemporal dynamics to traditional dynamics

A concrete example is the discrete time dynamical system known as *parallel update chip-firing* on a finite connected undirected graph $G = (V, E)$. The state of the system is a *chip configuration* $\mathbf{x} : V \to \mathbb{Z}$, and the time evolution is described by

$$\mathbf{x}_{t+1}(v) = \mathbf{x}_t(v) - d_v \mathbf{1}\{\mathbf{x}_t(v) \geq d_v\} + \sum_{u \sim v} \mathbf{1}\{\mathbf{x}_t(u) \geq d_u\},$$

where the sum is over the d_v neighbors u of vertex v. In words, at each discrete time step, each vertex v with at least as many chips as neighbors simultaneously *fires* by sending one chip to each of its neighbors.

For parallel update chip-firing on discrete torus graphs $\mathbb{Z}_n \times \mathbb{Z}_n$, Bagnolli et al. [3] plotted the average firing rate as a function of the total number of chips (placed independently at random to form the initial configuration $\mathbf{x}_0$). They discovered a mode-locking effect: Instead of increasing gradually, the firing rate remains constant over long intervals between which it increases sharply (Figure 1.1). The firing rate "likes" to be a simple rational number. This mode-locking has been proved in a special case, when G is a complete graph, by relating it to one of the canonical mode-locking systems, rotation number of a circle map [53].

Since $\sum_v \mathbf{x}_t(v)$ (the total number of chips) is conserved, only finitely many chip configurations are reachable from a given $\mathbf{x}_0$, and the sequence $(\mathbf{x}_t)_{t\geq 0}$ is eventually periodic. In practice one very often observes short periods. Exponentially long periods are possible on some graphs [51], but not on trees [10], cycles [28], complete bipartite [48] or complete graphs [53].

Periodic parallel chip-firing sequences are "nonclumpy": if some vertex fires twice in a row, then every vertex fires at least once in any two consecutive time steps [49].

Are mode-locking, short periods, and nonclumpiness inherent in the abelian network; or are they artifacts of the parallel update rule? In this work we find atemporal vestiges of some of these phenomena. For example, despite its definition involving parallel update, the firing rate is constant on components of the trajectory digraph (Proposition 6.6).

Abelian networks have the *confluence* property: any two legal executions are joinable. The Exchange Lemma 4.4 says that any two legal executions are joinable in the minimum possible number of steps. In the case of a critical network, we show that the number of additional steps needed is bounded from above by a constant that does not depend on the executions (Theorem 6.9).

1.4 Computational questions

Goles and Margenstern [39] showed that parallel update chip-firing on a suitably constructed infinite graph is capable of universal computation. The choice of parallel update is essential for the circuits in [39], which rely on the relative timing of signals along pairs of wires. Using the circuit designs of Moore and Nilsson [58], Cairns [20] proved that *regardless of the time parameterization*, chip-firing on the cubic lattice $\mathbb{Z}^3$ can emulate a Turing machine. Hence, even atemporal questions about chip-firing can be algorithmically undecidable. An example of such a question is: Given a triply periodic configuration of chips on $\mathbb{Z}^3$ plus finitely many additional chips, will the origin fire infinitely often?

What kinds of computation can be performed in a finite abelian network? In the atemporal viewpoint, a halting abelian network with k input wires and one output wire computes a function $f : \mathbb{N}^k \to \mathbb{N}$: If x_i chips are sent along the ith input wire for each $i = 1, \dots, k$, then regardless of the order in which the input chips arrive, exactly $f(x_1, \dots, x_k)$ chips arrive at the end of the output wire. Holroyd, Levine and Winkler [44] classify the functions f computable by a finite network of finite abelian processors: these are precisely the increasing functions of the form

$$f = L + P,$$

where L is a linear function with rational coefficients, and P is an eventually periodic function. Any such function can be computed by a finite halting abelian network of certain simple gates. An example that shows all gate types is

$$f(x, y, z) = \max(0, x-1) + \min(1, y) + \left\lfloor \frac{x + \lfloor 2z/3 \rfloor}{4} \right\rfloor .$$

The next subsections survey a few highlights of the dissertation. We have sacrificed some generality in order to state them with a minimum of notation.

The abelian network $\mathcal{N}$ in our main results is assumed to be finite and locally irreducible. We also assume that $\mathcal{N}$ is strongly connected for the latter half of the dissertation (Chapter 5-8).

1.5 The torsion group of a nonhalting abelian network

We are going to associate a finite abelian group $\mathrm{Tor}(\mathcal{N})$ to any finite, irreducible abelian network $\mathcal{N}$. In the case $\mathcal{N}$ is halting, $\mathrm{Tor}(\mathcal{N})$ coincides with the critical group of [17], which acts freely and transitively on the recurrent states of $\mathcal{N}$.

What does $\mathrm{Tor}(\mathcal{N})$ act on in the nonhalting case? Here it is more natural to work with weak connected components of the trajectory digraph. Sending input to $\mathcal{N}$ can shift it between components, and these shifts are quantified by the *shift monoid* $\mathcal{M}(\mathcal{N})$. The torsion group arises from the action of $\mathcal{M}(\mathcal{N})$ on the *invertible recurrent components* of the trajectory digraph. These are components that contain either a cycle or an infinite path, and such that the inverse action of $\mathcal{M}(\mathcal{N})$ on these components is well defined (see Definitions 4.8 and 4.19 for the details).

Now we can answer our motivating question about the dynamical significance of the group $\mathcal{G}$ defined in (1.1).

Theorem 1.1. *$\mathcal{G}$ is isomorphic to the Grothendieck group of the shift monoid $\mathcal{M}(\mathcal{N})$, and the torsion part of $\mathcal{G}$ acts freely on the invertible recurrent components of the trajectory digraph.*

Theorem 1.1 is proved in §4.3 as a corollary of Theorem 4.21. In the case that $\mathcal{N}$ is halting, the invertible recurrent components are in bijection with recurrent states, and this bijection preserves the group action (Theorem 4.28).

1.6 Critical networks

The critical networks (those with Perron-Frobenius eigenvalue $\lambda = 1$) are particularly interesting. They include sinkless chip-firing, rotor-routing, and their respective generalizations, arithmetical networks and agent networks (Figure 3.1).

A critical network has a conserved quantity which we call *level*; for example, the level of a chip-firing configuration is the total number of chips. We define the *capacity* of a critical network as the maximum level of a configuration that halts. A problem mentioned in [17] is to find algebraic invariants that can distinguish between "homotopic" networks (those with the same production matrix P and total kernel K). Capacity is such an invariant: Rotor and chip-firing networks on the same graph have the same P and K, but different capacities.

A halting network has recurrent *states*, and so far we have generalized this notion to recurrent *components* of the trajectory digraph. Can we choose a representative configuration in each component? In the halting case, yes: each recurrent component contains a unique configuration of the form $0.\mathbf{q}$ where $\mathbf{q}$ is a recurrent state. In a general nonhalting network it is not clear how to define recurrent *configurations* $\mathbf{x}.\mathbf{q}$. But we are able to define them in the critical case, and show that the recurrent components are precisely the components that contain a recurrent configuration. We then prove a recurrence test, Theorem 5.6, for configurations in a critical network, analogous to Dhar's burning test for states [30] (and Speer's extension of it to directed graphs, [64], further extended to halting networks in [17]). This answers another problem posed at the end of [17].

Our second main result for critical networks is a combinatorial description for the orbits of the action of the torsion group.

Theorem 1.2. *Let $\mathcal{N}$ be a critical network. Then for all but finitely many positive m the action of the torsion group on the recurrent components of level m*

$$\mathrm{Tor}(\mathcal{N}) \times \overline{\mathrm{Rec}}(\mathcal{N}, m) \to \overline{\mathrm{Rec}}(\mathcal{N}, m),$$

is free and transitive.

Theorem 1.2 is proved in §5.4 as a corollary of Theorem 5.25. The exceptional values of m are those for which there exists a halting configuration of level m.

1.7 Sandpile networks

An archetypal example of abelian networks is the sandpile network, which was first introduced by Dhar [30] to study the concept of self-organized criticality [4]. In this network, each vertex of the underlying digraph contains a finite number of chips. If a vertex has at least as many chips as its outgoing edges, then we are allowed to *fire* the vertex by sending one chip along each of its outgoing edges to the neighbors of the vertex. This network is abelian in the sense that, although there will often be many possible choices for the order in which to fire vertices, the final configuration (if the network is halting) does not depend on the chosen order.

Sandpile networks come with two flavors, one with a vertex chosen as a sink and one without any sinks. In the former case the chips ending at the sink are removed from the network, and as a result the corresponding network is halting. In the latter case no chips are ever removed from the network, and as a result the corresponding network is nonhalting.

Despite belonging to different families of abelian networks, the dynamics of those two networks are nevertheless very similar to each other, and our first result for sandpile networks is an identification between recurrent states of the sanpdile network with a sink and recurrent components of sinkless sandpile networks.

Theorem 1.3. *Let G be a strongly connected Eulerian digraph, and let s be the vertex chosen as the sink. Then, for any $m \geq 0$, the number of recurrent components of the sinkless sandpile network that have exactly m chips is equal to the number of recurrent states of the sandpile network with a sink that have most $m - \operatorname{outdeg}(s)$ chips.*

We prove Theorem 1.3 by showing that the following procedure is a bijection: given any recurrent component, find a configuration where the sink is the only vertex than can be fired, and then remove an appropriate amount of chips from the sink to get the desired recurrent state (see Proposition 7.12).

When the underlying digraph G is substituted with a bidirected graph (i.e., when the reverse of every edge is also an edge), the dynamics of the sandpile network can be related to the *Tutte polynomial* [69] of G. This is the polynomial in two variables given by

$$T_G(x,y) := \sum_{A \subseteq E(G)} (x-1)^{k(A)-k(E(G))}(y-1)^{k(A)+|A|-|V(G)|},$$

where $k(A)$ denotes the number of connected components of the graph $(V(G), A)$. Tutte polynomial admits several different equivalent definitions, including one involving deletion-contraction recurrence, one involving activities of spanning trees, and one involving the random cluster model [37] from statistical physics. See [69] for more details. (Note that $T_G(x,y)$ is not defined for general directed graphs!).

The relation between sandpile networks and the Tutte polynomial was first proved by Merino López [57], who showed that

$$\sum_{m\geq 0} a_m y^m = \frac{T_G(1,y)}{y^{|E(G)|/2}}\left(y + y^2 + \ldots + y^{\mathrm{outdeg}(s)}\right), \tag{1.2}$$

where a_m is the number of recurrent states with m chips. (The right hand side of (1.2) contains an extra factor that does not appear in [57] due to a difference in what constitutes a sink; see Example 3.14.) The identity (1.2) was used to prove a conjecture of Stanley [65] on the h-vector of a matroid complex in the special case of a cographic matroid, and the original proof of (1.2) relied on the deletion-contraction recurrence of the Tutte polynomial. A bijective proof of this identity connecting recurrent states and undirected spanning trees was later given by Cori and Le Borgne [26]. Our second main result, Theorem 7.4, is the extension of Cori-Le Borgne bijection when G is an Eulerian digraph, where undirected spanning trees are replaced by arborescences (directed spanning trees oriented away from a sink vertex) . As a consequence, we establish an identity analogous to (1.2) but with $T_G(1,y)$ being replaced with a single variable generalization of the Tutte polynomial called the greedoid polynomial [11]. This answers another problem posed by Perrot and Pham [59]. (We remark that the greedoid polynomial is defined for *all* digraphs, but Theorem 7.4 is true only for *Eulerian* digraphs. This is because the identity (1.2) never depends on the choice of the sink, while the greedoid polynomial is independent of the sink only if G is Eulerian.)

1.8 Rotor networks and abelian mobile agents

The critical networks of zero capacity (i.e., those that run forever on any positive input) are precisely the "abelian mobile agents" defined in [15] (see Lemma 8.8).

In particular these include the *sinkless rotor networks*, whose defining property is that each vertex serves its neighbors in a fixed periodic order. The walk performed by a single chip input to a sinkless rotor network has variously been called ant walk [70], Eulerian walk [62], rotor walk [45], quasirandom rumor spreading [33], and "deterministic random walk" [25].

Let $G = (V, E)$ be a finite, strongly connected directed graph with multiple edges permitted. For each vertex v, fix a cyclic permutation t_v of the outgoing edges from v. The role of t_v is to specify the order in which v serves its neighbors.

A *chip-and-rotor configuration* is a pair $\mathbf{x}.\rho$, where $\mathbf{x} : V \to \mathbb{Z}$ indicates the number of chips at each vertex, and $\rho : V \to E$ assigns an outgoing edge to each vertex. The legal moves in a sinkless rotor network are as follows: For a vertex v such that $\mathbf{x}(v) \geq 1$, replace $\rho(v)$ by $\rho'(v) := t_v(\rho(v))$, and then transfer one chip from v to the other endpoint of $\rho'(v)$.

A *cycle* of ρ is a minimal nonempty set of vertices $C \subset V$ such that $\rho(v) \in C$ for all $v \in C$. Tóthmérész [67, Theorem 2.4] proved the following test for recurrence; the special case when $\mathbf{x}$ has just one chip goes back to [43, Theorem 3.8].

Theorem 1.4 (Cycle test for recurrence in a sinkless rotor network, [67]). *A chip-and-rotor configuration* $\mathbf{x}.\rho$ *is recurrent if and only if* $\mathbf{x} \in \mathbb{N}^V$ *and* $\sum_{v \in C} \mathbf{x}(v) \geq 1$ *for every cycle* C *of* ρ.

For the general statement when G is not strongly connected, see [67, Theorem 2.4]. In §8.1 we present a new proof of Theorem 1.4 that extends to all abelian mobile agent networks (see Theorem 8.4 for the detail).

The proof of Theorem 1.4 uses an idea that is also present in the proof of Theorem 1.3 relating sandpile networks with a sink to its sinkless counterpart, which is in turn based on an idea of Levine [54]. One motivation for this work is to see how far this technique can be generalized. To that end, we introduce *thief networks*, which are halting networks constructed from a given critical network. We show that the recurrent configurations of an agent network can be determined from the recurrent states of its thief networks, and vice versa (Lemma 8.12).

Using the cycle test, it becomes a problem of pure combinatorics to enumerate the recurrent chip-and-rotor configurations. Their generating function has the following determinantal form.

Theorem 1.5. *For $n \geq 1$, let r_n be the number of recurrent chip-and-rotor configurations with exactly n chips on a finite, strongly connected digraph G. Then we have the following identity (in $\mathbb{C}$ for $|z| < 1$, and also in the ring of formal power series $\mathbb{Z}[[z]]$):*

$$\sum_{n \geq 1} r_n z^n = \det\left(\frac{D_G}{1-z} - A_G\right),$$

where D_G is the diagonal matrix of outdegrees, and A_G is the adjacency matrix of G.

In particular, it follows from Theorem 1.5 that the sequence $(r_n)_{n \geq 1}$ determines the characteristic polynomial of the Markov transition matrix $(AD^{-1})^\top$ for random walk on G. A multivariate version (in $\#V + \#E$ variables) of Theorem 1.5 is given in Theorem 8.11.

As an application, we use Theorem 1.5 to give an alternative proof for the classical *matrix tree theorem* [68]. This theorem states that the number of reverse arborescenes of G rooted toward vertex v is equal to the cofactor of the *Laplacian matrix* $D_G - A_G$ corresponding to v. This theorem has inspired several other results that enumerate combinatorial objects by a determinantal formula, which includes formulas that enumerate cycle-rooted spanning forests [36, 50], formulas that compute edge correlations of random spanning forests [19], and our own Theorem 1.5. Our proof of the matrix tree theorem is included in Appendix A.

1.9 Removal lemma

A basic tool underlying many of our results is the Removal Lemma 4.2, which extends both the exchange lemma of Björner, Lovász, and Shor [13] and the least action principle [35, 15]. It implies that if m is the minimal length of a periodic path in the trajectory digraph of a (finite, irreducible) critical abelian network, then any periodic path can be shortened to a periodic path of length m, and any two periodic paths of length m have the same multiset of edge labels. One could view this fact as an atemporal version of the short period phenomenon described in §1.2.

The rest of the dissertation is organized as follows: In Chapter 2 we discuss the relevant commutative monoid theory that used to construct the torsion group. In Chapter 3 we review the theory of halting abelian networks from [15, 16, 17]. In Chapter 4, Chapter 5, Chapter 6, and Chapter 8 we prove the theorems in §1.5, §1.6, §1.3 and §1.8, respectively.

1.10 Summary of notation

$\mathcal{M}$	a commutative monoid
$\mathcal{K}$	the Grothendieck group of $\mathcal{M}$

$\tau(\mathcal{K})$	the torsion subgroup of $\mathcal{K}$
$X^{\times}$	the set of $\tau(\mathcal{K})$-invertible elements of X (Def. 2.2)
$\mathcal{F}$	a finite commutative monoid
e	the minimal idempotent of $\mathcal{F}$ (Def. 2.6)
$G = (V, E)$	a directed graph
A_G	the adjacency matrix of G
D_G	the outdegree matrix of G
Δ	the Laplacian matrix of G
Δ_s	the reduced Laplacian matrix of G at a vertex s
$\mathcal{P}_v$	the processor at vertex v (§3.1)
A_v	the input alphabet of $\mathcal{P}_v$ (§3.1)
Q_v	the state space of $\mathcal{P}_v$ (§3.1)
$\mathcal{N}$	an abelian network (§3.1)
A	the total alphabet of $\mathcal{N}$ (§3.1)
Q	the total state space of $\mathcal{N}$ (§3.1)
A^*	the free monoid on A
$\mathbb{N}$	the set $\{0, 1, 2, \dots\}$ of nonnegative integers
$\mathbf{0}$	the vector in $\mathbb{Z}^A$ with all entries equal to 0
$\mathbf{1}$	the vector in $\mathbb{Z}^A$ with all entries equal to 1
$\mathbf{m}, \mathbf{n}$	vectors in $\mathbb{N}^A$
$\mathbf{x}, \mathbf{y}, \mathbf{z}$	vectors in $\mathbb{Z}^A$
$\mathbf{x}^+, \mathbf{x}^-$	the positive and negative part of $\mathbf{x} \in \mathbb{Z}^A$
w	a word in the alphabet A
$\lvert w\rvert$	the vector in $\mathbb{N}^A$ counting the multiplicity of each letter in w
T_v	the transition function of vertex v (§3.1)
$T_{(v,u)}$	the message passing function of edge (v, u) (§3.1)

$t_w(\mathbf{q})$	the state after $\mathcal{N}$ in state $\mathbf{q}$ processes w (§3.1)
$\mathbf{M}_w(\mathbf{q})$	the message passing vector of w and $\mathbf{q}$ (§3.1)
$\mathbf{p}, \mathbf{q}$	states in Q
$\mathbf{x}.\mathbf{q}$	a configuration of $\mathcal{N}$ (§3.1)
$\pi_w(\mathbf{x}.\mathbf{q})$	the configuration $(\mathbf{x} + \mathbf{M}_w(\mathbf{q}) - \lvert w \rvert).t_w(\mathbf{q})$
$\mathbf{x}.\mathbf{q} \overset{w}{\dashrightarrow} \mathbf{x}'.\mathbf{q}'$	w is an execution from $\mathbf{x}.\mathbf{q}$ to $\mathbf{x}'.\mathbf{q}'$ (§3.2)
$\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$	w is a legal execution from $\mathbf{x}.\mathbf{q}$ to $\mathbf{x}'.\mathbf{q}'$ (§3.2)
$\mathbf{x}.\mathbf{q} \dashrightarrow \mathbf{x}'.\mathbf{q}'$	there exists an execution from $\mathbf{x}.\mathbf{q}$ to $\mathbf{x}'.\mathbf{q}'$
$\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$	there exists a legal execution from $\mathbf{x}.\mathbf{q}$ to $\mathbf{x}'.\mathbf{q}'$
$\mathrm{Loc}(\mathcal{N})$	locally recurrent states of $\mathcal{N}$ (§3.3)
K	the total kernel of $\mathcal{N}$ (Def. 3.6)
P	the production matrix of $\mathcal{N}$ (Def. 3.8)
$\lambda(P)$	the spectral radius of P
$\mathrm{supp}(\mathbf{x})$	the set $\{a \in A \mid \mathbf{x}(a) \neq 0\}$
$w \setminus \mathbf{n}$	the removal of $\mathbf{n}$ from w (§4.1)
$\mathbf{x}.\mathbf{q} \dashrightarrow\dashleftarrow \mathbf{x}'.\mathbf{q}'$	$\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ are quasi-legally related (Def. 4.6)
$\mathbf{x}.\mathbf{q} \to\leftarrow \mathbf{x}'.\mathbf{q}'$	$\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ are legally related (Def. 4.6)
$\overline{\mathbf{x}.\mathbf{q}}$	the equivalence class for $\to\leftarrow$ that contains $\mathbf{x}.\mathbf{q}$
$\overline{\mathrm{Rec}}(\mathcal{N})$	the set of recurrent components of $\mathcal{N}$ (Def. 4.8)
$\mathcal{M}(\mathcal{N})$	the shift monoid of $\mathcal{N}$ (Def. 4.17)
$\mathcal{K}(\mathcal{N})$	the Grothendieck group of $\mathcal{N}$ (§4.3)
$\mathrm{Tor}(\mathcal{N})$	the torsion group of $\mathcal{N}$ (Def. 4.18)
$\overline{\mathrm{Rec}}(\mathcal{N})^{\times}$	the set of invertible recurrent components of $\mathcal{N}$ (Def. 4.19)
$\mathcal{S}$	a subcritical abelian network
$\mathcal{F}(\mathcal{S})$	the global monoid of $\mathcal{S}$ (§4.4)
$\mathbf{r}$	the period vector of $\mathcal{N}$ (Def. 5.1)

$\mathcal{N}_R$	the thief network on $\mathcal{N}$ restricted to $R \subseteq A$ (§5.2)
$\mathbf{1}_R$	the indicator vector for $R \subseteq A$ in $\mathbb{Z}^A$
$\mathbf{x}_R$	the vector in $\mathbb{Z}^A$ given by $\mathbf{x}_R(\cdot) := \mathbf{1}_R(\cdot)\mathbf{x}(\cdot)$
$\mathbf{s}$	the exchange rate vector of $\mathcal{N}$ (Def. 5.13)
cap	the capacity of an object (Def. 5.14)
lvl	the level of an object (Def. 5.17)
$\overline{\mathrm{Rec}}(\mathcal{N}, m)$	the set of recurrent components with level m
$\mathrm{Stop}(\mathcal{N})$	the set of stoppable levels of $\mathcal{N}$ (Def. 5.21)
$\mathbf{h}$	The vector $(I - P_R)\mathbf{r}$
$\mathbb{Z}_0^A$	the set $\{\mathbf{z} \in \mathbb{Z}^A \mid \mathbf{s}^\top \mathbf{z} = 0\}$
$\varrho_{\mathbf{q}}$	the rotor digraph of $\mathbf{q}$ (Def. 8.1)
M_R	the $A \times A$ matrix $(\mathbf{1}_R(a) M(a, a'))_{a,a' \in A}$
$\mathrm{Rec}(\mathcal{N}, \mathbf{n})$	the set of recurrent configurations with input $\mathbf{n}$
$\mathrm{Rec}(\mathcal{N}, m)$	the set of recurrent configurations with level m

CHAPTER 2

COMMUTATIVE MONOID ACTIONS

In this chapter we review some commutative monoid theory that will be used in Chapter 4 to construct the torsion group of an abelian network. Parts of this material are covered in greater generality in [40, 52, 41, 66].

2.1 Injective actions and Grothendieck group

Let $\mathcal{M}$ be a *commutative monoid*, i.e., a set equipped with an associative and commutative operation $(m, n) \mapsto mn$ with an identity element $\epsilon \in \mathcal{M}$ satisfying $\epsilon m = m$ for all $m \in \mathcal{M}$.

The *Grothendieck group* $\mathcal{K}$ of $\mathcal{M}$ is $(\mathcal{M} \times \mathcal{M})/\sim$, where $(m_1, m_1') \sim (m_2, m_2')$ if there is $m \in \mathcal{M}$ such that $mm_1m_2' = mm_1'm_2$. The multiplication of $\mathcal{K}$ is defined coordinate-wise. The set $\mathcal{K}$ is an abelian group under this operation.

The Grothendieck group satisfies the *universal enveloping property*: If $f : \mathcal{M} \to H$ is a monoid homomorphism into an abelian group H, then there exists a unique group homomorphism $f_* : \mathcal{K} \to H$ such that the following diagram commutes:

$$\begin{tikzcd} \mathcal{M} \arrow[r, "f"] \arrow[d, "\iota"] & H \\ \mathcal{K} \arrow[ur, "f_*"] & \end{tikzcd} ,$$

where $\iota : \mathcal{M} \to \mathcal{K}$ is the map $m \mapsto \overline{(m, \epsilon)}$.

An *action* of a monoid $\mathcal{M}$ on a set X is an operation $(m, x) \mapsto mx$ such that $\epsilon x = x$ and $m(m'x) = (mm')x$ for all $x \in X$ and $m, m' \in \mathcal{M}$.

Definition 2.1 (Injective action). Let $\mathcal{M}$ be a commutative monoid. An action of $\mathcal{M}$ on X is *injective* if, for all $x, x' \in X$ and all $m \in \mathcal{M}$, we have that $mx = mx'$ implies $x = x'$. $\triangle$

Definition 2.2 (Invertible element). Let $\mathcal{M}$ be a commutative monoid that acts on X. Let H be a subgroup of the Grothendieck group $\mathcal{K}$ of $\mathcal{M}$. An element $x \in X$ is *H-invertible* if, for any $g \in H$, there exists $x_g \in X$ such that

$$mx = m'x_g,$$

for any representative (m, m') of g. We denote by X_H the set of H-invertible elements of X. (Note that the set X_H can be empty for some choice of H.) $\triangle$

For any subgroup H of $\mathcal{K}$, we define the group action of H on X_H by

$$\begin{aligned} H \times X_H &\to X_H \\ (g, x) &\mapsto x_g, \end{aligned}$$

where x_g is as in Definition 2.2. In the next lemma we show that this is a well-defined group action if $\mathcal{M}$ acts injectively on X.

Lemma 2.3. *Let $\mathcal{M}$ be a commutative monoid that acts injectively on X, and let H be a subgroup of the Grothendieck group $\mathcal{K}$ of $\mathcal{M}$. For any $g \in H$ and any H-invertible element x,*

(i) *The corresponding element x_g is unique.*

(ii) *The element x_g is H-invertible.*

(iii) *For any $h \in H$, we have $h(gx) = (hg)x$.*

Proof. (i) Let (m, m') be a representative of g and let $x_1, x_2 \in X_H$ be such that $mx = m'x_1$ and $mx = m'x_2$. This implies that $m'x_1 = mx = m'x_2$. Since $\mathcal{M}$ acts injectively on X, this implies that $x_1 = x_2$. This completes the proof.

(ii) Let h be an arbitrary element of H and (n, n') an arbitrary representative of h. Let x_{hg} be an element of X such that $nmx = n'm'x_{hg}$. Note that x_{hg} exists because x is H-invertible and $hg = \overline{(nm, n'm')} \in H$. Then

$$m'n'x_{hg} = n'm'x_{hg} = nmx = nm'x_g = m'nx_g.$$

Since $\mathcal{M}$ acts injectively on X, the equation above implies that $n'x_{hg} = nx_g$. Since the choice of h and (n, n') are arbitrary, the claim now follows.

(iii) Let $(n, n') \in h$ and $x_{hg} \in X$ be such that $nmx = n'm'x_{hg}$. It suffices to show that x_{hg} satisfies $nx_g = n'x_{hg}$, and note that this has been done in the proof of part (ii). $\square$

The action of $\mathcal{M}$ on X is *free* if, for any $x \in X$ and $m, m' \in \mathcal{M}$, we have $mx = m'x$ implies that $m = m'$.

Lemma 2.4. *Let $\mathcal{M}$ be a commutative monoid that acts injectively on X, and let H be a subgroup of the Grothendieck group $\mathcal{K}$ of $\mathcal{M}$.*

(i) *If $\mathcal{M}$ acts freely on X, then H acts freely on X_H.*

(ii) *If H is finite and X is nonempty, then X_H is nonempty.*

Proof. (i) Suppose that $\overline{(m_1, m'_1)}, \overline{(m_2, m'_2)} \in H$ and $x \in X_H$ are such that $\overline{(m_1, m'_1)}x = \overline{(m_2, m'_2)}x$. Then

$$\begin{aligned} & m_1 m'_2 x = m'_1 m_2 x && \text{(by Definition 2.2)} \\ \Longrightarrow\quad & m_1 m'_2 = m'_1 m_2 && \text{(because } \mathcal{M} \text{ acts freely on } X) \\ \Longrightarrow\quad & \overline{(m_1, m'_1)} = \overline{(m_2, m'_2)} && \text{(by the definition of Grothendieck group).} \end{aligned}$$

This proves the claim.

(ii) Let $g_1, \dots, g_k$ be an enumeration of the elements of H. For each $i \in \{1, \dots, k\}$, choose a representative (m_i, m_i') of g_i, and write $m_H := m_1' \cdots m_k'$. Since X is nonempty, the set $m_H X$ is also nonempty. Hence it suffices to show that $m_H X \subseteq X_H$.

For any $i \in \{1, \dots, k\}$ and any $x \in X$, write $x_i := m_i m_1' \cdots \widehat{m_i'} \cdots m_k' x$. Then

$$m_i m_H x = m_i m_1' \cdots m_k' x = m_i' m_i m_1' \cdots \widehat{m_i'} \cdots m_k' x = m_i' x_i, \tag{2.1}$$

by the commutativity of the monoid.

Let i be an arbitrary element of $\{1, \dots, k\}$, and let (n_i, n_i') be an arbitrary representative of g_i. Since (m_i, m_i') and (n_i, n_i') are contained in g_i, there exists $m \in \mathcal{M}$ such that $m m_i n_i' = m m_i' n_i$. Then, continuing from (2.1),

$$\begin{aligned} & m_i m_H x = m_i' x_i \quad \Longrightarrow \quad m n_i' m_i m_H x = m n_i' m_i' x_i \\ \Longrightarrow \quad & m m_i n_i' m_H x = m m_i' n_i' x_i \quad \Longrightarrow \quad m m_i' n_i m_H x = m m_i' n_i' x_i \\ \Longrightarrow \quad & n_i m_H x = n_i' x_i \qquad \text{(because } \mathcal{M} \text{ acts injectively).} \end{aligned}$$

Since the choice of i and (n_i, n_i') are arbitrary, it then follows from Definition 2.2 that $m_H x$ is H-invertible. $\square$

Let $\tau(\mathcal{K})$ be the *torsion subgroup* of $\mathcal{K}$,

$$\tau(\mathcal{K}) := \{g \in \mathcal{K} \mid g \text{ has finite order}\}.$$

The monoid $\mathcal{M}$ is *finitely generated* if there is a finite subset A of $\mathcal{M}$ such that every $m \in \mathcal{M}$ can be written as a product of finitely many elements in A. Note that $\tau(\mathcal{K})$ is a finite group if $\mathcal{M}$ is finitely generated by the fundamental theorem of finitely generated abelian groups. We denote by $X^\times$ the set of $\tau(\mathcal{K})$-invertible elements of X.

The following proposition is a corollary of Lemma 2.4.

Proposition 2.5. *Let $\mathcal{M}$ be a finitely generated commutative monoid that acts freely and injectively on a nonempty set X. Then $X^{\times}$ is a nonempty set; and $\tau(\mathcal{K})$ is a finite abelian group that acts freely on $X^{\times}$.* $\square$

2.2 The case of finite commutative monoids

Here we refine the results of the previous section to the case when the monoid is finite.

Let $\mathcal{F}$ be a finite commutative monoid that acts on a set Y.

Definition 2.6 (Minimal idempotent). The *minimal idempotent* of a finite commutative monoid $\mathcal{F}$ is

$$e := \prod_{f \in \mathcal{F}, ff = f} f. \qquad \triangle$$

Equivalently, the minimal idempotent is the unique element of $\mathcal{F}$ satisfying $ee = e$ and $e \in m\mathcal{F}$ for any $m \in \mathcal{F}$ (see [16]).

The action of $\mathcal{F}$ on Y is *irreducible* if for any $y, y' \in Y$ there exist $m, m' \in \mathcal{F}$ such that $my = m'y'$.

Lemma 2.7 ([16, Lemma 2.2, Lemma 2.3, Lemma 2.4]). *Let $\mathcal{F}$ be a finite commutative monoid that acts on Y, and let e be the minimal idempotent of $\mathcal{F}$.*

(i) *The set $e\mathcal{F}$ is a finite abelian group with identity element e.*

(ii) *If the action of $\mathcal{F}$ on Y is irreducible and $y \in eY$, then for any $y' \in Y$ there exists $m' \in \mathcal{F}$ such that $m'y' = y$.*

(iii) *For every $m \in \mathcal{F}$, the map defined by $y \mapsto my$ is a bijection from eY to eY.* $\square$

Let $X := eY$, and let $\eta : \mathcal{F} \to \mathrm{End}(X)$ be the (monoid) homomorphism induced by the action of $\mathcal{F}$ on X. We denote by $\mathcal{M}$ the image of $\mathcal{F}$ under the map η. Just like in §2.1, we denote by $\mathcal{K}$ the Grothendieck group of $\mathcal{M}$, and by $X^\times$ the set of $\tau(\mathcal{K})$-invertible elements of X.

The action of $\mathcal{F}$ on Y is *faithful* if there do not exist distinct $m, m' \in \mathcal{F}$ such that $my = m'y$ for all $y \in Y$. A set $Y' \subseteq Y$ is *closed* under the action of $\mathcal{F}$ if $mY' \subseteq Y'$ for all $m \in \mathcal{F}$.

Proposition 2.8. *Let $\mathcal{F}$ be a finite commutative monoid that acts faithfully and irreducibly on a nonempty set Y, and let $X := eY$. Then*

(i) *X is the unique nonempty closed subset of Y on which $\mathcal{F}$ acts injectively.*

(ii) *The group $e\mathcal{F}$ is isomorphic to $\tau(\mathcal{K})$ by the map $\varphi : e\mathcal{F} \to \tau(\mathcal{K})$ defined by $em \mapsto \overline{(\eta(em), \epsilon)}$.*

(iii) *$X^\times$ is equal to X.*

(iv) *The isomorphism $\varphi : e\mathcal{F} \to \tau(\mathcal{K})$ preserves the action of $e\mathcal{F}$ and $\tau(\mathcal{K})$ on $X = X^\times$.*

Proof. (i) The set X is closed since $mX = m(eY) = e(mY) = eY = X$ by commutativity. The set X is nonempty since Y is nonempty. By Lemma 2.7(iii), the action of $\mathcal{F}$ on $X = eY$ is injective.

Suppose that X' is another nonempty closed subset of Y such that $\mathcal{F}$ acts injectively on X'. Let x' be an arbitrary element of X'. Note that $ex' = eex'$ since e is an idempotent. The injectivity assumption then implies that $x' = ex'$. This shows that $X' \subseteq eY = X$.

Let y be any element of Y, and let x' be an element of X' (note that x' exists because X' is nonempty). By the irreducibility assumption, there exist $m, m' \in \mathcal{F}$

such that $my = m'x'$. Applying Lemma 2.7(ii) to $ey \in eY$ and $my \in Y$, there exists $m'' \in \mathcal{F}$ such that $m''my = ey$. Hence we have

$$ey = m''my = m''m'x'.$$

Now note that $m''m'x'$ is in X' since X' is closed. Since the choice of y is arbitrary, we conclude that $X = eY \subseteq X'$. This proves the claim.

(ii) We first show that the map η sends $e\mathcal{F}$ to $\mathcal{M}$ bijectively. Note that the action of e on $eY = X$ is trivial as e is idempotent, and hence $\eta(e)$ is the identity element of $\mathcal{M}$. Then

$$\eta(e\mathcal{F}) = \eta(e)\eta(\mathcal{F}) = \mathcal{M},$$

which shows surjectivity. For injectivity, let $m, m' \in \mathcal{F}$ be such that $\eta(em) = \eta(em')$. Then

$$em(ey) = em'(ey) \quad \forall y \in Y \quad \Longrightarrow \quad emy = em'y \quad \forall y \in Y.$$

Since the action of $\mathcal{F}$ on Y is faithful, the equation above implies that $em = em'$. This shows injectivity.

Since $e\mathcal{F}$ is a finite group by Lemma 2.7(i) and $\eta : e\mathcal{F} \to \mathcal{M}$ is a bijective monoid homomorphism, we conclude that $\mathcal{M}$ is a finite group and η is a group isomorphism. Since $\mathcal{M}$ is a group, the map $\iota : \mathcal{M} \to \mathcal{K}$ is a group isomorphism by the universal enveloping property of Grothendieck group. Since $\mathcal{M}$ is finite, we have the group $\mathcal{K}$ is finite, and hence $\mathcal{K} = \tau(\mathcal{K})$. Now note that

$$\begin{tikzcd} e\mathcal{F} \arrow[r, "\eta"] \arrow[rr, bend right, "\varphi"'] & \mathcal{M} \arrow[r, "\iota"] & \mathcal{K} = \tau(\mathcal{K}) \end{tikzcd}.$$

Since η and ι are group isomorphisms, it follows that φ is a group isomorphism, as desired.

(iii) Since $\mathcal{M}$ is a group, all elements of X are $\tau(\mathcal{K})$-invertible, as desired.

(iv) This follows from the definition of η. $\square$

CHAPTER 3

ABELIAN NETWORKS

Here we recall the basic setup of abelian networks, referring the reader to [15, 16] for the details. Sinkless rotor and sinkless sandpile networks (Examples 3.11 and 3.12) are the basic examples to keep in mind when reading this chapter.

3.1 Definition of abelian networks

Let $G = (V(G), E(G))$ be a directed graph (or a *digraph* for short), which may have self-loops and multiple edges. We will write V and E instead of $V(G)$ and $E(G)$ if the digraph G is evident from the context. An *outgoing edge* of v is an edge with source vertex v, and the *outdegree* $\mathrm{outdeg}(v)$ of v is the number of outgoing edges of v. We denote by $\mathrm{Out}(v)$ the set of outgoing edges of v. An *out-neighbor* of v is the target vertex of an outgoing edge of v. The *indegree* and the *in-neighbors* of v are defined analogously.

In an *abelian network* $\mathcal{N}$ with underlying digraph G, each vertex $v \in V$ has a *processor* $\mathcal{P}_v$, which is an automaton with (nonempty) input alphabet A_v and (nonempty) state space Q_v. The data specifying the automaton are:

(i) A *transition function* $T_a : Q_v \to Q_v$ for each $a \in A_v$; and

(ii) A *message-passing function* $T_e : Q_v \times A_v \to A_u^*$ for each edge e directed from v to u,

where A_u^* denotes the free monoid of all finite words in the alphabet A_u. In the event that the processor $\mathcal{P}_v$ in state $q \in Q_v$ processes a letter $a \in A_v$, the automaton transitions to the state $T_a(q)$ and sends the message $T_e(q,a)$ to $\mathcal{P}_u$ for each edge e directed from v to u.

We require these functions to satisfy commutativity conditions, i.e., for any $a, b \in A_v$ and any $q \in Q_v$,

(i) $T_a \circ T_b = T_b \circ T_a$; and

(ii) For any outgoing edge e of v, the word $T_e(q,a)T_e(T_a(q),b)$ is equal to $T_e(q,b)T_e(T_b(q),a)$ up to permuting the letters.

Described in words, permuting the letters processed by $\mathcal{P}_v$ does not change the resulting state of the processor $\mathcal{P}_v$, and may change the output sent to $\mathcal{P}_u$ only by permuting its letters.

The *(total) state space* is $Q := \prod_{v \in V} Q_v$, and the *(total) alphabet* is $A := \sqcup_{v \in V} A_v$. An *input* of $\mathcal{N}$ is a vector $\mathbf{x} \in \mathbb{Z}^A$, where $\mathbf{x}(a)$ indicates the number of a's that are waiting to be processed (here we allow negative entries in $\mathbf{x}$ for a technical reason that will be apparent soon). A *state* $\mathbf{q}$ of $\mathcal{N}$ is an element of the total state space Q, where $\mathbf{q}(v)$ indicates the state of the processor $\mathcal{P}_v$. A *configuration* of $\mathcal{N}$ is a pair $\mathbf{x}.\mathbf{q}$, where $\mathbf{x}$ is an input and $\mathbf{q}$ is a state of $\mathcal{N}$.

Let $a \in A$, and let $v \in V$ be such that $a \in A_v$. The *(total) transition function* $t_a : Q \to Q$ is given by

$$t_a\mathbf{q}(u) := \begin{cases} T_a(\mathbf{q}(u)) & \text{if } u = v; \\ \mathbf{q}(u) & \text{otherwise.} \end{cases}$$

(Note that we write $t_a\mathbf{q}$ instead of $t_a(\mathbf{q})$ to simplify the notation.) The *message-*

passing vector $\mathbf{M}_a : Q \to \mathbb{N}^A$ is given by

$$\mathbf{M}_a(\mathbf{q}) := \sum_{e \in \mathrm{Out}(v)} |T_e(\mathbf{q}(v), a)|,$$

where $|w|$ is the vector in $\mathbb{N}^A$ such that $|w|(a)$ is the number of a's in the word w ($a \in A$). (We adopt the convention that $\mathbb{N}$ denotes the set $\{0, 1, \ldots\}$ of nonnegative integers.) Described in words, $\mathbf{M}_a(\mathbf{q})(b)$ is the number of b's produced when network $\mathcal{N}$ in state $\mathbf{q}$ processes the letter a.

In the event that $\mathcal{N}$ processes a copy of the letter a on the configuration $\mathbf{x}.\mathbf{q}$, the following three things happen:

(i) The state of $\mathcal{N}$ changes to $t_a\mathbf{q} \in Q$;

(ii) $\mathbf{M}_a(\mathbf{q})(b)$ many b's are created for each $b \in A$; and

(iii) The processed letter a is removed from $\mathcal{N}$.

This process can be described formally by the *configuration transition function* $\pi_a : \mathbb{Z}^A \times Q \to \mathbb{Z}^A \times Q$, given by

$$\pi_a(\mathbf{x}.\mathbf{q}) := (\mathbf{x} + \mathbf{M}_a(\mathbf{q}) - |a|).t_a\mathbf{q}.$$

We extend the transition functions defined above to any finite word $w = a_1 \ldots a_\ell$ over A by:

$$\begin{aligned}
t_w\mathbf{q} &:= t_{a_\ell} \cdots t_{a_1}\mathbf{q}, \\
\mathbf{M}_w(\mathbf{q}) &:= \sum_{i=1}^{\ell} \mathbf{M}_{a_i}(t_{a_{i-1}} \cdots t_{a_1}\mathbf{q}), \\
\pi_w(\mathbf{x}.\mathbf{q}) &:= \pi_{a_\ell} \cdots \pi_{a_1}(\mathbf{x}.\mathbf{q}) = (\mathbf{x} + \mathbf{M}_w(\mathbf{q}) - |w|).t_w\mathbf{q},
\end{aligned}$$

which encode the state, the generated letters, and the configuration obtained after processing the word w, respectively.

For any $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^A$, we write $\mathbf{x} \leq \mathbf{y}$ if $\mathbf{y} - \mathbf{x}$ is a vector with nonnegative entries.

Lemma 3.1 ([15, Lemma 4.1, Lemma 4.2]). *Let $\mathcal{N}$ be an abelian network, and let $w, w' \in A^*$.*

(i) (Monotonicity) *If $|w| \leq |w'|$, then $\mathbf{M}_w(\mathbf{q}) \leq \mathbf{M}_{w'}(\mathbf{q})$ for all $\mathbf{q} \in Q$.*

(ii) (Abelian property) *If $|w| = |w'|$, then $t_w = t_{w'}$, $\pi_w = \pi_{w'}$, and $\mathbf{M}_w = \mathbf{M}_{w'}$.* □

Lemma 3.1(ii) implies that the functions t_w, π_w, and $\mathbf{M}_w$ depend only on the vector $|w|$. Therefore, we can extend these transition functions to any vector $\mathbf{w} \in \mathbb{N}^A$ by setting

$$t_{\mathbf{w}} := t_w, \qquad \pi_{\mathbf{w}} := \pi_w, \qquad \mathbf{M}_{\mathbf{w}} := \mathbf{M}_w,$$

where w is any word such that $\mathbf{w} = |w|$.

3.2 Legal and complete executions

An *execution* is a word $w \in A^*$, which prescribes an order in which the letters in w are to be processed. We assume that an execution is finite, unless stated otherwise.

Let $w = a_1 \cdots a_\ell$, and let $\mathbf{x}.\mathbf{q}$ be a configuration of $\mathcal{N}$. We write $\mathbf{x}_i.\mathbf{q}_i := \pi_{a_i} \cdots \pi_{a_1}(\mathbf{x}.\mathbf{q})$ for $i \in \{0, 1, \ldots, \ell\}$. We say that w is a *legal execution* for $\mathbf{x}.\mathbf{q}$ if $\mathbf{x}_{i-1}(a_i) \geq 1$ for all $i \in \{1, \ldots, \ell\}$. We say that w is a *complete execution* for $\mathbf{x}.\mathbf{q}$ if $\mathbf{x}_\ell(a) \geq 0$ for all $a \in A$.

Definition 3.2 ($\dashrightarrow$ and $\longrightarrow$). Let $\mathcal{N}$ be an abelian network. We write $\mathbf{x}.\mathbf{q} \overset{w}{\dashrightarrow} \mathbf{x}'.\mathbf{q}'$ if $\pi_w(\mathbf{x}.\mathbf{q}) = \mathbf{x}'.\mathbf{q}'$. We write $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$ if $\pi_w(\mathbf{x}.\mathbf{q}) = \mathbf{x}'.\mathbf{q}'$ and w is a legal execution for $\mathbf{x}.\mathbf{q}$. △

In order to simplify the notation, we will write $\dashrightarrow$ and $\longrightarrow$ when the word w is not a major component of the discussion. We remark that $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}.\mathbf{q}$ since the empty word is a legal execution that sends $\mathbf{x}.\mathbf{q}$ to $\mathbf{x}.\mathbf{q}$.

In the next lemma, we list several properties of $\dashrightarrow$ and $\longrightarrow$. The *support* of a vector $\mathbf{u} \in \mathbb{Z}^A$ is $\operatorname{supp}(\mathbf{u}) := \{a \in A \mid \mathbf{u}(a) \neq 0\}$.

Lemma 3.3. *Let $\mathcal{N}$ be an abelian network.*

(i) *If $\mathbf{x}.\mathbf{q} \overset{w}{\dashrightarrow} \mathbf{x}'.\mathbf{q}'$, then $(\mathbf{x}+\mathbf{z}).\mathbf{q} \overset{w}{\dashrightarrow} (\mathbf{x}'+\mathbf{z}).\mathbf{q}'$ for all $\mathbf{z} \in \mathbb{Z}^A$.*

(ii) *If $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$ and $\mathbf{z} \in \mathbb{Z}^A$ satisfies $\mathbf{z}(a) \geq 0$ for all $a \in \operatorname{supp}(|w|)$, then $(\mathbf{x}+\mathbf{z}).\mathbf{q} \xrightarrow{w} (\mathbf{x}'+\mathbf{z}).\mathbf{q}'$.*

(iii) *For any $a \in A$, if $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$ and $|w|(a) > 0$, then $\mathbf{x}'(a) \geq 0$.*

(iv) *If $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$ and $\mathbf{x}'.\mathbf{q}' \xrightarrow{w'} \mathbf{x}''.\mathbf{q}''$, then $\mathbf{x}.\mathbf{q} \xrightarrow{ww'} \mathbf{x}''.\mathbf{q}''$.*

Proof. This follows directly from the definition of $\dashrightarrow$ and $\longrightarrow$. $\square$

3.3 Locally recurrent states

An abelian network $\mathcal{N}$ is *finite* if both the (total) state space Q and the (total) alphabet A are finite sets. All abelian networks in this dissertation are assumed to be finite, unless stated otherwise.

We denote by $M \subseteq \operatorname{End}(Q)$ the *transition monoid* $\langle t_a \rangle_{a \in A}$. Note that M is a finite commutative monoid as $\mathcal{N}$ is finite. Since M is finite, it has a (unique) minimal idempotent e (Definition 2.6).

A state $\mathbf{q} \in Q$ is *locally recurrent* if $\mathbf{q} \in eQ$. We denote by $\mathrm{Loc}(\mathcal{N})$ the set of locally recurrent states of $\mathcal{N}$. For maximum generality we don't assume local recurrence, but the reader will not lose much by restricting the state space of the network to $\mathrm{Loc}(\mathcal{N})$. Note that $\mathrm{Loc}(\mathcal{N})$ is a nonempty set (since Q is nonempty by definition of $\mathcal{N}$).

Here we list properties of locally recurrent states that will be used in this dissertation. We denote by $\mathbf{1}$ the vector $(1, \ldots, 1)^\top$ in $\mathbb{Z}^A$.

Lemma 3.4. *Let $\mathcal{N}$ be a finite abelian network. Then*

(i) *There exists $\mathbf{e} \in \mathbb{N}^A$ such that $t_\mathbf{e}\mathbf{q}$ is locally recurrent for all $\mathbf{q} \in Q$.*

(ii) *A state $\mathbf{q}$ is locally recurrent if there exists $\mathbf{n} \in \mathbb{N}^A$ such that $\mathbf{n} \geq \mathbf{1}$ and $t_\mathbf{n}\mathbf{q} = \mathbf{q}$.*

Proof. (i) The claim follows by taking $\mathbf{e}$ to be a vector in $\mathbb{N}^A$ such that $t_\mathbf{e}$ is the minimal idempotent of M.

(ii) Since $\mathbf{n} \geq \mathbf{1}$, we can without loss of generality assume that $t_\mathbf{n} \in eM$ (by taking a finite multiple of $\mathbf{n}$ if necessary). Then $\mathbf{q} = t_\mathbf{n}\mathbf{q} \in t_\mathbf{n}Q \subseteq eQ$, and hence $\mathbf{q}$ is locally recurrent. $\square$

Lemma 3.5. *Let $\mathcal{N}$ be a finite abelian network. For any $\mathbf{n} \in \mathbb{N}^A$,*

(i) *The function $t_\mathbf{n}$ restricted to $\mathrm{Loc}(\mathcal{N})$ is a bijection from $\mathrm{Loc}(\mathcal{N})$ to $\mathrm{Loc}(\mathcal{N})$.*

(ii) *The function $\pi_\mathbf{n}$ restricted to $\mathbb{Z}^A \times \mathrm{Loc}(\mathcal{N})$ is a bijection from $\mathbb{Z}^A \times \mathrm{Loc}(\mathcal{N})$ to $\mathbb{Z}^A \times \mathrm{Loc}(\mathcal{N})$.*

Proof. The first part of the lemma follows directly from Lemma 2.7(iii). The second part of the lemma is a consequence of the first part. $\square$

3.4 The production matrix

For any vector $\mathbf{z} \in \mathbb{Z}^A$, the *positive part* $\mathbf{z}^+$ and *negative part* $\mathbf{z}^-$ of $\mathbf{z}$ are the unique vectors in $\mathbb{N}^A$ such that $\mathbf{z} = \mathbf{z}^+ - \mathbf{z}^-$ and $\operatorname{supp}(\mathbf{z}^+) \cap \operatorname{supp}(\mathbf{z}^-) = \varnothing$.

Definition 3.6 (Total kernel). Let $\mathcal{N}$ be a finite abelian network. The *total kernel* $K \subseteq \mathbb{Z}^A$ is

$$K := \{\mathbf{z} \in \mathbb{Z}^A \mid t_{\mathbf{z}^+}\mathbf{q} = t_{\mathbf{z}^-}\mathbf{q} \text{ for all } \mathbf{q} \in \operatorname{Loc}(\mathcal{N})\}. \qquad \triangle$$

We say that $\mathcal{N}$ is *locally irreducible* if for any $\mathbf{q}, \mathbf{q}' \in Q$ there exist $w, w' \in A^*$ such that $t_w\mathbf{q} = t_{w'}\mathbf{q}'$.

Lemma 3.7 ([16, Lemma 4.5, Lemma 4.6]). *Let $\mathcal{N}$ be a finite abelian network.*

(i) *The total kernel K is a subgroup of $\mathbb{Z}^A$ of finite index.*

(ii) *If $\mathcal{N}$ is locally irreducible, then for any $\mathbf{q} \in \operatorname{Loc}(\mathcal{N})$,*

$$K \cap \mathbb{N}^A = \{\mathbf{x} \in \mathbb{N}^A \mid t_{\mathbf{x}}\mathbf{q} = \mathbf{q}\}. \qquad \square$$

For $\mathbf{q} \in \operatorname{Loc}(\mathcal{N})$, we define $P_{\mathbf{q}} : K \cap \mathbb{N}^A \to \mathbb{Z}^A$ to be

$$P_{\mathbf{q}}(\mathbf{k}) := \mathbf{M}_{\mathbf{k}}(\mathbf{q}).$$

The map $P_{\mathbf{q}}$ extends uniquely to a group homomorphism $K \to \mathbb{Z}^A$ [16, Lemma 4.6]. Since K is a subgroup of $\mathbb{Z}^A$ of finite index (by Lemma 3.7(i)), we get a linear map $P_{\mathbf{q}} : \mathbb{Q}^A \to \mathbb{Q}^A$ by tensoring the group homomorphism $P_{\mathbf{q}}$ with $\mathbb{Q}$.

If $\mathcal{N}$ is locally irreducible, then the matrix $P_{\mathbf{q}} : \mathbb{Q}^A \to \mathbb{Q}^A$ does not depend on the choice of $\mathbf{q}$ [16, Lemma 4.9].

Definition 3.8 (Production matrix). Let $\mathcal{N}$ be a finite and locally irreducible abelian network. The *production matrix* of $\mathcal{N}$ is the matrix $P := P_{\mathbf{q}}$, where $\mathbf{q}$ is any locally recurrent state of $\mathcal{N}$. △

Lemma 3.9. *Let $\mathcal{N}$ be a finite and locally irreducible abelian network. If $\mathbf{q} \in Q$ and $\mathbf{n}, \mathbf{n}' \in \mathbb{N}^A$ satisfy $t_{\mathbf{n}}\mathbf{q} = t_{\mathbf{n}'}\mathbf{q}$, then*

$$\mathbf{n} - \mathbf{n}' \in K \qquad \textit{and} \qquad \mathbf{M}_{\mathbf{n}}(\mathbf{q}) - \mathbf{M}_{\mathbf{n}'}(\mathbf{q}) = P(\mathbf{n} - \mathbf{n}').$$

Proof. By Lemma 3.4(i), there exists $\mathbf{e} \in \mathbb{N}^A$ such that $\mathbf{p} := t_{\mathbf{e}}\mathbf{q}$ is locally recurrent. Write $\mathbf{p}' := t_{\mathbf{n}}\mathbf{p} = t_{\mathbf{n}'}\mathbf{p}$. Since $\mathcal{N}$ is locally irreducible and $\mathbf{p} \in \mathrm{Loc}(\mathcal{N})$, by Lemma 2.7(ii) there exists $\mathbf{m} \in \mathbb{N}^A$ such that $t_{\mathbf{m}}\mathbf{p}' = \mathbf{p}$.

By the abelian property (Lemma 3.1(ii)), we have:

$$\begin{tikzcd} \mathbf{q} \arrow[r, bend left, "t_{\mathbf{n}}"] \arrow[r, bend right, "t_{\mathbf{n}'}"'] \arrow[d, "t_{\mathbf{e}}"] & \mathbf{q}' \arrow[d, "t_{\mathbf{e}}"] & \\ \mathbf{p} \arrow[r, bend left, "t_{\mathbf{n}}"] \arrow[r, bend right, "t_{\mathbf{n}'}"'] & \mathbf{p}' \arrow[r, "t_{\mathbf{m}}"] & \mathbf{p}. \end{tikzcd} \quad . \tag{3.1}$$

In particular, the bottom row of Diagram (3.1) above gives us $t_{\mathbf{n}+\mathbf{m}}\mathbf{p} = t_{\mathbf{n}'+\mathbf{m}}\mathbf{p} = \mathbf{p}$. By Lemma 3.7(ii) these equations imply that both $\mathbf{n} + \mathbf{m}$ and $\mathbf{n}' + \mathbf{m}$ are in K, and hence $\mathbf{n} - \mathbf{n}' \in K$.

By the abelian property and the commutativity of Diagram (3.1),

$$\mathbf{M}_{\mathbf{n}}(\mathbf{q}) + \mathbf{M}_{\mathbf{e}+\mathbf{m}}(\mathbf{q}') = \mathbf{M}_{\mathbf{e}}(\mathbf{q}) + \mathbf{M}_{\mathbf{n}+\mathbf{m}}(\mathbf{p});$$

$$\mathbf{M}_{\mathbf{n}'}(\mathbf{q}) + \mathbf{M}_{\mathbf{e}+\mathbf{m}}(\mathbf{q}') = \mathbf{M}_{\mathbf{e}}(\mathbf{q}) + \mathbf{M}_{\mathbf{n}'+\mathbf{m}}(\mathbf{p}).$$

By subtracting one equation from the other,

$$\mathbf{M}_{\mathbf{n}}(\mathbf{q}) - \mathbf{M}_{\mathbf{n}'}(\mathbf{q}) = \mathbf{M}_{\mathbf{n}+\mathbf{m}}(\mathbf{p}) - \mathbf{M}_{\mathbf{n}'+\mathbf{m}}(\mathbf{p}).$$

Since $\mathbf{n}+\mathbf{m}$ and $\mathbf{n}'+\mathbf{m}$ are in K and $\mathbf{p} \in \mathrm{Loc}(\mathcal{N})$,

$$\mathbf{M}_{\mathbf{n}+\mathbf{m}}(\mathbf{p}) - \mathbf{M}_{\mathbf{n}'+\mathbf{m}}(\mathbf{p}) = P(\mathbf{n}+\mathbf{m}) - P(\mathbf{n}'+\mathbf{m}) = P(\mathbf{n}-\mathbf{n}').$$

This completes the proof. □

3.5 Subcritical, critical, and supercritical abelian networks

Let $\mathcal{N}$ be a finite and locally irreducible abelian network. The *production digraph* Γ is the directed graph with vertex set A and edge set $\{(a,b) : P_{ba} > 0\}$.

We define an equivalence relation on A by considering a and b to be equivalent if there exists a directed path from a to b and a directed path from b to a in Γ. The *strong components* of Γ are the equivalence classes of this relation. A network $\mathcal{N}$ is *strongly connected* if Γ has only one strong component.

The *spectral radius* of the production matrix P is

$$\lambda(P) := \max\{|\lambda| \,:\, \lambda \text{ is an eigenvalue of } P\}.$$

We distinguish (finite, locally irreducible) abelian networks by the value of $\lambda(P)$:

- The network $\mathcal{N}$ is *subcritical* if $\lambda(P) < 1$. Subcritical networks are studied in [16, 17].
- The network $\mathcal{N}$ is *critical* if $\lambda(P) = 1$. We will study critical networks in more detail in the latter half of this dissertation.
- The network $\mathcal{N}$ is *supercritical* if $\lambda(P) > 1$.

See Example 3.17 for a concrete example of each network.

Let $A_1, \ldots, A_s$ be the strong components of Γ. Denote by P_i the matrix obtained by restricting the production matrix P to rows and columns from A_i. We say that A_i is a *subcritical* component if $\lambda(P_i) < 1$, and a letter $a \in A$ is *subcritical* if it is contained in a subcritical component. *Critical* and *supercritical* components/letters are defined analogously.

We denote by $A_<$ the set of subcritical letters, and by $A_\leq$ the set of subcritical and critical letters. The sets $A_=$, $A_>$, and $A_\geq$ are defined analogously. Recall that the *support* of $\mathbf{u} \in \mathbb{R}^A$ is $\operatorname{supp}(\mathbf{u}) := \{a \in A \mid \mathbf{u}(a) \neq 0\}$.

A real matrix P is *nonnegative* if all its entries are nonnegative, and is *positive* if all of its entries are positive. For all matrices P and Q of the same dimension, we write $Q \leq P$ if $P - Q$ is a nonnegative matrix. *Nonnegative vectors* and *positive vectors* are defined analogously.

We now present variants of the Perron-Frobenius theorem that will be used in this dissertation, referring to [7] for most of the proof.

Lemma 3.10 (Perron-Frobenius). *Let A be a finite set, and let P be an $A \times A$ matrix whose entries are nonnegative rational numbers.*

(i) *P has a nonnegative real eigenvector with eigenvalue $\lambda(P)$.*

(ii) *If α is a real number such that $P\mathbf{u} = \alpha\mathbf{u}$ for some positive vector $\mathbf{u} \in \mathbb{R}^A$, then $\alpha = \lambda(P)$.*

(iii) *Let P be strongly connected, and let α be a real number such that $P\mathbf{u} \geq \alpha\mathbf{u}$ for some nonzero nonnegative vector $\mathbf{u} \in \mathbb{R}^A$. Then $\lambda(P) \geq \alpha$, and equality holds if and only if $P\mathbf{u} = \alpha\mathbf{u}$. Furthermore, the claim is still true if "$\geq$" is replaced with "$\leq$".*

(iv) *If P is strongly connected and Q is a nonnegative matrix such that $Q \leq P$ and $Q \neq P$, then $\lambda(Q) < \lambda(P)$.*

(v) *If P is strongly connected, then the eigenspace of $\lambda(P)$ is spanned by a positive real vector.*

(vi) *If P is strongly connected and $\lambda(P) \in \mathbb{Q}$, then the eigenspace of $\lambda(P)$ is spanned by a positive integer vector.*

(vii) *There exists $\mathbf{n}, \mathbf{n}', \mathbf{n}'' \in \mathbb{N}^A$ such that*

- $\operatorname{supp}(\mathbf{n}) = A_<$ *and* $P\mathbf{n}(a) < \mathbf{n}(a)$ *for all* $a \in A_<$;
- $\operatorname{supp}(\mathbf{n}') = A_=$ *and* $P\mathbf{n}'(a) \geq \mathbf{n}'(a)$ *for all* $a \in A_=$; *and*
- $\operatorname{supp}(\mathbf{n}'') = A_>$ *and* $P\mathbf{n}''(a) > \mathbf{n}''(a)$ *for all* $a \in A_>$.

(viii) *There exists* $\mathbf{m} \in \mathbb{N}^A$ *such that* $\operatorname{supp}(\mathbf{m}) = A_\geq$ *and* $P\mathbf{m}(a) \geq \mathbf{m}(a)$ *for all* $a \in A_\geq$.

Proof. (i) This follows from [7, Theorem 2.1.1].

(ii) This follows from [7, Theorem 2.1.11].

(iii) This follows from [7, Theorem 2.1.11].

(iv) This follows from [7, Theorem 2.1.5(b)].

(v) This follows from [7, Theorem 2.1.4(b)].

(vi) Since both P and $\lambda(P)$ are rational, the eigenspace $\mathcal{E}$ of $\lambda(P)$ has a basis that consists of integer vectors. It then follows from part (v) that $\mathcal{E}$ is spanned by a positive integer vector.

(vii) We prove only the subcritical case, as the other two cases are analogous. Let $A_1, \ldots, A_k$ be the subcritical components of Γ. Write $\lambda_i := \lambda(P_i)$ ($i \in \{1, \ldots, k\}$). Note that $\lambda_i < 1$ by assumption.

It follows from part (v) that for each $i \in \{1, \ldots, k\}$ there exists a nonnegative vector $\mathbf{u}_i \in \mathbb{R}^A$ such that $\mathrm{supp}(\mathbf{u}_i) = A_i$ and $P\mathbf{u}_i(a) = \lambda_i \mathbf{u}_i(a)$ for all $a \in A_i$. By scaling and rounding $\mathbf{u}_i$ if necessary, there exist $\mathbf{n}_i \in \mathbb{N}^A$ and sufficiently small $\epsilon_i > 0$ such that $\mathrm{supp}(\mathbf{n}_i) = A_i$ and $P\mathbf{n}_i(a) < (1-\epsilon_i)\mathbf{n}_i(a)$ for all $a \in A_i$. By scaling $\mathbf{n}_1, \ldots, \mathbf{n}_k$ if necessary, we can assume that $\mathbf{n} := \mathbf{n}_1 + \ldots + \mathbf{n}_k$ satisfies $P\mathbf{n}(a) < \mathbf{n}(a)$ for all $a \in A_< = A_1 \sqcup \ldots \sqcup A_k$. This proves the lemma.

(viii) Let $\mathbf{m} := \mathbf{n}' + \mathbf{n}''$, where $\mathbf{n}'$ and $\mathbf{n}''$ are as in part (vii). Then for any critical letter a,

$$P\mathbf{m}(a) = P\mathbf{n}'(a) + P\mathbf{n}''(a) \geq P\mathbf{n}'(a) \geq \mathbf{n}'(a) = \mathbf{m}(a),$$

and for any supercritical letter a,

$$P\mathbf{m}(a) = P\mathbf{n}'(a) + P\mathbf{n}''(a) \geq P\mathbf{n}''(a) > \mathbf{n}''(a) = \mathbf{m}(a).$$

This proves the lemma. $\square$

Remark. We would like to warn the reader that the subcritical variant of part (viii) (i.e., there exists $\mathbf{m} \in \mathbb{N}^A$ such that $\mathrm{supp}(\mathbf{m}) = A_\leq$ and $P\mathbf{m}(a) \leq \mathbf{m}(a)$ for all $a \in A_\leq$) is false. Indeed, let P be the matrix

$$P = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

A direct computation then shows that the inequality $P\mathbf{m} \leq \mathbf{m}$ is always false for any positive vector $\mathbf{m}$.

3.6 Examples: sandpiles, rotor, toppling, etc

In this section we present several examples of abelian networks. The relationship between these networks is illustrated in Figure 3.1.

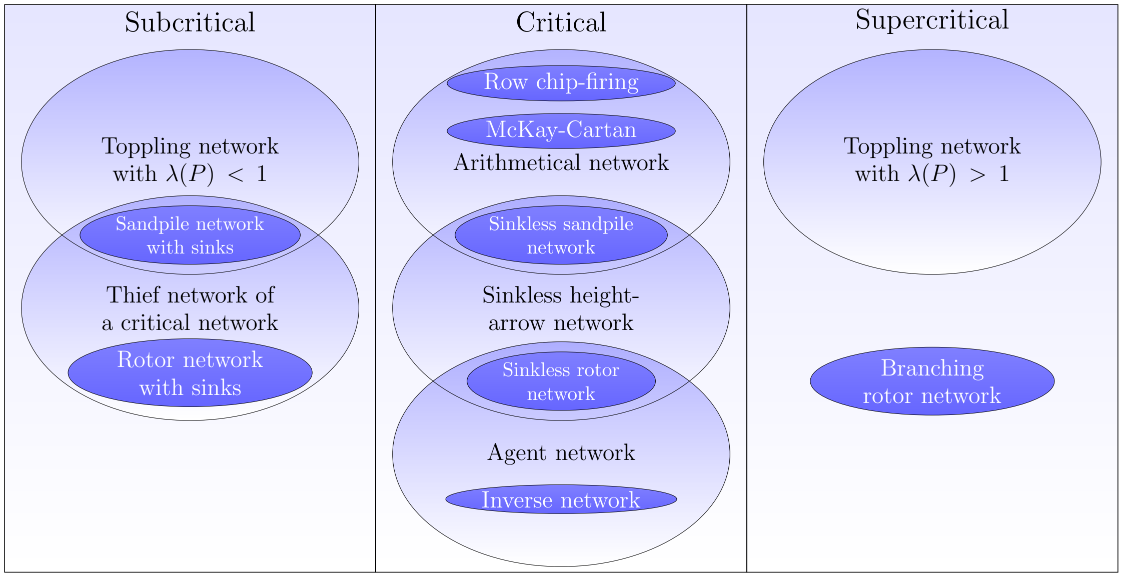

Figure 3.1: A Venn diagram illustrating several classes of (finite and locally irreducible) abelian networks. The Perron-Frobenius eigenvalue λ increases from left to right. In the middle bubble in the critical column, the capacity (see Definition 5.14) increases from bottom to top.

We use the following graph theory terminology throughout this dissertation. Recall that G is a directed graph with vertex set V and edge set E. A digraph is *Eulerian* if for all $v \in V$ the outdegree of v is equal to the indegree of v. Any undirected graph can be changed into an Eulerian directed graph by replacing each undirected edge $\{v, u\}$ with a pair of directed edges (v, u) and (u, v). We call such a digraph *bidirected*.

The *adjacency matrix* A_G of G is the matrix $(a_{v,v'})_{v,v' \in V}$, where $a_{v,v'}$ is the number of edges directed from v' to v. The *outdegree matrix* D_G of G is the $V \times V$ diagonal matrix with $D_G(v, v) := \text{outdeg}(v)$ $(v \in V)$. The *Laplacian matrix* L_G of G is the matrix $D_G - A_G$.

The digraph G is *strongly connected* if for any $v, v' \in V$ there exists a directed path in G from v to v'.

The following digraph will be our main running example for the underlying digraph of an abelian network. For $n \geq 3$, the *bidirected cycle* C_n is

$$V(C_n) := \{v_k \mid k \in \mathbb{Z}_n\}, \qquad E(C_n) := \bigcup_{k \in \mathbb{Z}_n} \{(v_k, v_{k-1}), (v_k, v_{k+1})\}.$$

All networks presented in this section are *irreducible*, i.e. they satisfy these two properties:

- The network is locally irreducible; and
- The minimal idempotent of the transition monoid M is the identity element of M.

In particular, any state of an irreducible network is locally recurrent.

Example 3.11 (Sinkless rotor network [62, 70, 63]). For each vertex $v \in V$, fix a cyclic total order on the set of the outgoing edges $\mathrm{Out}(v)$ of v, i.e. an enumeration $e_0^v, e_1^v, \ldots, e_{\mathrm{outdeg}(v)-1}^v$ indexed by $\mathbb{Z}_{\mathrm{outdeg}(v)}$. The alphabet, state space, and state transition of the processor $\mathcal{P}_v$ are given by

$$A_v := \{v\}, \qquad Q_v := \mathrm{Out}(v), \qquad T_v(e_i^v) := e_{i+1}^v \quad (i \in \mathbb{Z}_{\mathrm{outdeg}(v)}).$$

For each edge e_j^v directed from v to u_j^v, the message-passing function is given by

$$T_{e_j^v}(e_i^v, v) := \begin{cases} u_j^v & \text{if } i = j-1; \\ \epsilon & \text{otherwise.} \end{cases}$$

A state of the full network is described by a *rotor configuration* of G, that is, a function $V \to E$ assigning to each vertex v an outgoing edge from v. When a

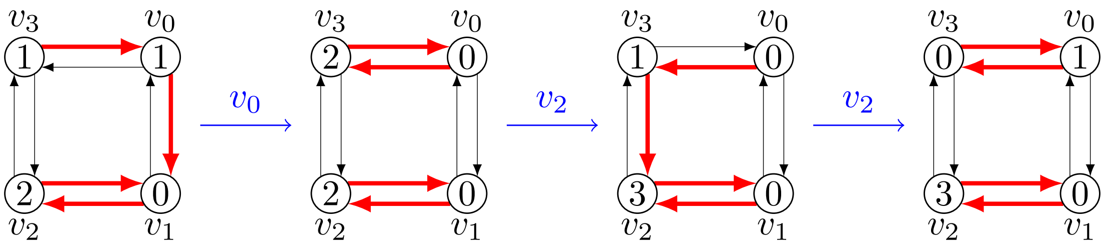

Figure 3.2: A three-step legal execution in the sinkless rotor network on the bidirected cycle C_4. The number on each vertex records the number of letters waiting to be processed, and the (red) thick outgoing edge records the state of the processor.

chip/letter at vertex v is processed, the edge/state e_i^v assigned to v changes to e_{i+1}^v (the next edge in the cyclic total order), and the processed chip is moved from v to the target vertex of e_{i+1}^v. See Figure 3.2 for an illustration of the process.

Any sinkless rotor network is strongly connected if the underlying digraph G is strongly connected. The total kernel and the production matrix of this network are given by

$$K = \{\mathbf{z} \in \mathbb{Z}^V \mid \mathbf{z}(v) \text{ is divisible by } \operatorname{outdeg}(v) \text{ for all } v \in V\}; \qquad P = A_G D_G^{-1},$$

where A_G is the adjacency matrix of G and D_G is the outdegree matrix of G. Because $\mathbf{1} A_G D_G^{-1} = \mathbf{1}$, the Perron-Frobenius theorem (Lemma 3.10(ii)) implies that $\lambda(P) = 1$. Hence this network is a critical network. $\triangle$

Example 3.12 (Sinkless sandpile network/chip-firing [30, 13]). For each vertex $v \in V$ of the underlying digraph, the processor $\mathcal{P}_v$ is given by

$$A_v := \{v\}, \qquad Q_v := \{0, 1, \ldots, \operatorname{outdeg}(v) - 1\}, \qquad T_v(i) := i + 1 \bmod \operatorname{outdeg}(v).$$

For each edge e directed from v to u, the message-passing function is given by

$$T_e(i, v) := \begin{cases} u & \text{if } i = \operatorname{outdeg}(v) - 1; \\ \epsilon & \text{otherwise.} \end{cases}$$

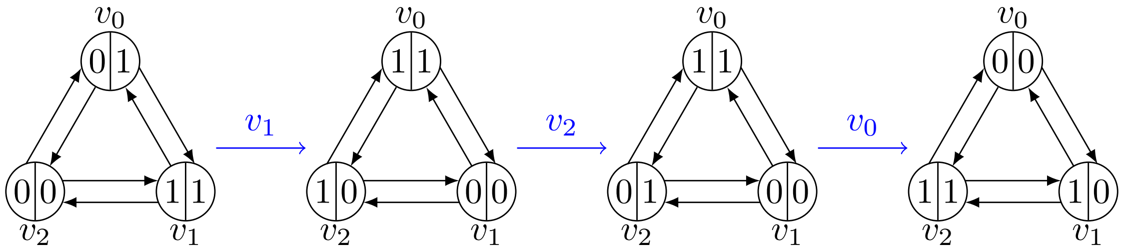

Figure 3.3: A three-step legal execution in the sinkless sandpile network on the bidirected cycle C_3. In the figure, the left part of a vertex records the number of letters waiting to be processed, and the right part records the state of the processor.

We can think of each processor $\mathcal{P}_v$ as a "locker" that can store up to $\text{outdeg}(v)-1$ chips, and its state q_v represents the number of chips it is currently storing. When $\mathcal{P}_v$ receives a new chip, the chip is stored in the locker if it has unallocated space (i.e., if $\mathbf{q}(v) < \text{outdeg}(v) - 1$). If the locker is already full (i.e., $\mathbf{q}(v) = \text{outdeg}(v) - 1$), then $\mathcal{P}_v$ sends all $\text{outdeg}(v) - 1$ stored chips plus the extra chip to its neighbors by sending one chip along each outgoing edge from v. See Figure 3.3 for an illustration of this process.

The total kernel and the production matrix of this network are equal to the corresponding objects in the sinkless rotor network (with the same underlying digraph). Hence by the same reasoning as Example 3.11, a sinkless sandpile network on a strongly connected digraph is a critical network.

Remark. We would like to warn the reader that (network) configurations in this dissertation have a subtle difference when compared to (chip) configurations in the literature. A (chip) configuration in the usual sense is a vector $\mathbf{c} \in \mathbb{Z}^V$ that records the number of chips at each vertex. By contrast, a (network) configuration in this dissertation is a pair $\mathbf{x}.\mathbf{q}$, where the vector $\mathbf{x} \in \mathbb{Z}^V$ records the number of chips that are not stored in the lockers, and the state $\mathbf{q} \in \prod_{v \in V} \mathbb{Z}_{\text{outdeg}(v)}$ records the number of chips currently stored in the lockers.

Identifying $\mathbb{Z}_{\mathrm{outdeg}(v)}$ with $\{0, 1, \ldots, \mathrm{outdeg}(v) - 1\}$, the chip configuration corresponding to $\mathbf{x}.\mathbf{q}$ is the vector sum $\mathbf{x} + \mathbf{q}$. In particular, there is more than one way to represent a chip configuration as a network configuration. $\triangle$

Example 3.13 (Sinkless height-arrow network [29]). In this network, each vertex $v \in V$ of the underlying digraph G is assigned *threshold value* $\tau_v \in \{1, \ldots, \mathrm{outdeg}(v)\}$. The processor $\mathcal{P}_v$ is given by

$$\begin{aligned} A_v :=& \{v\}, \\ Q_v :=& \{(d, c) \in \{0, \ldots, \mathrm{outdeg}(v) - 1\} \times \{0, \ldots, \tau_v - 1\} \mid \\ & \quad d \equiv k\tau_v \pmod{\mathrm{outdeg}(v)} \text{ for some } k \in \mathbb{Z}\}, \end{aligned}$$

$$T_v(d, c) := \begin{cases} (d, c + 1) & \text{if } c < \tau_v - 1; \\ (d + \tau_v \bmod \mathrm{outdeg}(v), 0) & \text{if } c = \tau_v - 1. \end{cases}$$

For each $v \in V$, fix a cyclic total order $\{e^v_j \mid j \in \mathbb{Z}_{\mathrm{outdeg}(v)}\}$ on the set of outgoing edges of v. The message-passing function for the edge e^v_j directed from v to u^v_j is given by

$$T_{e^v_j}(d, c, v) := \begin{cases} u^v_j & \text{if } c = \tau_v - 1 \text{ and } j - d \in \{1, \ldots, \tau_v\} \pmod{\mathrm{outdeg}(v)}; \\ \epsilon & \text{otherwise.} \end{cases}$$

For each $v \in V$, the state (d, c) of $\mathcal{P}_v$ represents an arrow pointing from v to u^v_d, and with c chips sitting on v. When the vertex v collects τ_v chips, the arrow is incremented τ_v times, and one chip is sent to each vertex in $\{u^v_{d+j} \mid 1 \leq j \leq \tau_v\}$. See Figure 3.4 for an illustration of this process.

Note that sinkless rotor networks are height-arrow networks with $\tau_v = 1$ for all $v \in V$, and sinkless sandpile networks are height-arrow networks with $\tau_v = \mathrm{outdeg}(v)$ for all $v \in V$.

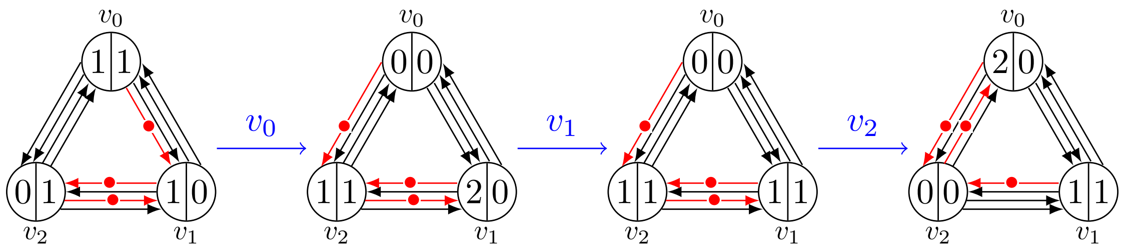

Figure 3.4: A three-step legal execution in a sinkless height-arrow network. For every $v \in V$, the threshold τ_v is equal to 2, and the cyclic total order on $\mathrm{Out}(v)$ is the counterclockwise ordering. In the figure, the left part of a vertex records the number of letters waiting to be processed, the right part records the height c_v of the processor, and the marked (red) outgoing edge records the arrow d_v of the processor.

Height-arrow networks have the same total kernel and production matrix as sinkless rotor and sandpile networks. In particular, height-arrow networks on a strongly connected digraph are critical networks.

Remark. Note that height-arrow networks as originally defined in [29] have state space $Q_v = \mathbb{Z}_{\mathrm{outdeg}(v)} \times \mathbb{Z}_{\tau_v}$ instead. Their choice of state space is in general not locally irreducible, and our choice of Q_v restricts the state space to an irreducible component of the network. $\triangle$

Example 3.14 (Height-arrow network with sinks). Fix a nonempty set $S \subseteq V$ that we designate as *sinks*. For each $v \in V$, assign a threshold value $\tau_v \in \{1, \ldots, \mathrm{outdeg}(v)\}$ and a cyclic total order $\{e_j^v \mid j \in \mathbb{Z}_{\mathrm{outdeg}(v)}\}$ to the out-going edges of v.

The alphabet A_v, the state space Q_v, and the transition function T_v are the same as in sinkless height-arrow networks. The message-passing function for the

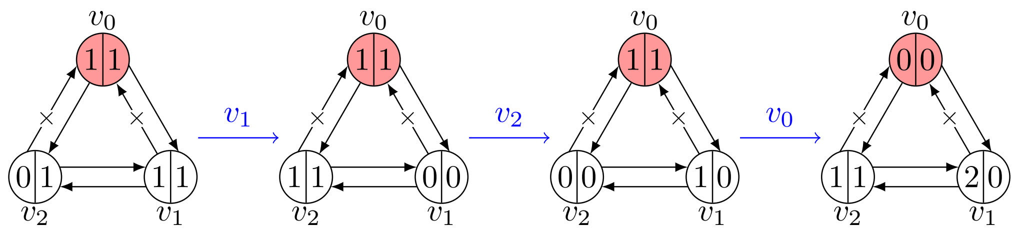

Figure 3.5: A three-step legal execution in the sandpile network with a sink at $S = \{v_0\}$. The incoming edges of a sink are marked with "$\times$". (Note that the left part of $v \in V$ records $\mathbf{x}(v)$, while the right part records $\mathbf{q}(v)$.)

edge e_j^v directed from v to u_j^v is given by

$$T_{e_j^v}(d, c, v) := \begin{cases} u_j^v & \text{if } c = \tau_v - 1,\ j - d \in \{1, \ldots, \tau_v\} \pmod{\operatorname{outdeg}(v)}, \text{ and } v_j \notin S; \\ \epsilon & \text{otherwise.} \end{cases}$$

This network is identical to the sinkless height-arrow network, except that letters passing through any edge pointing to the sink are removed from the network. See Figure 3.5 for an illustration of this process.

The total kernel of a height-arrow network with sinks is equal to the total kernel of the corresponding sinkless height-arrow network. The production matrix P of this network is equal to the matrix $A_G D_G^{-1}$ with rows corresponding to S replaced with zero vectors. Since $P \leq A_G D_G^{-1}$ and $\lambda(A_G D_G^{-1}) = 1$, we have by the Perron-Frobenius theorem (Lemma 3.10(iv)) that $\lambda(P) < 1$ (if G is strongly connected). Hence a height-arrow network with sinks on a strongly connected digraph is subcritical, unlike its sinkless counterpart.

Remark. In [15] a sink is defined as a processor with one state that sends no messages. However, in this dissertation we follow the convention from [23] that places sinks on the *incoming edges* to each $s \in S$ instead. The user can still opt to send input to s, and the processor $\mathcal{P}_s$ can still send messages to its out-neighbors. This

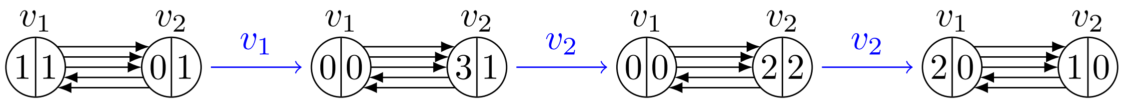

Figure 3.6: A three-step legal execution in a row chip-firing network (i.e., $d_{v_1} = 2$ and $d_{v_2} = 3$). In the figure, the left part of a vertex records the number of letters waiting to be processed, and the right part records the state of the processor.

extra flexibility comes in handy when we relate critical and subcritical networks in §5.2. △

Example 3.15 (Arithmetical network [55]). This network is determined by the pair $(\mathcal{D}, \mathbf{b})$, where $\mathcal{D}$ is a diagonal matrix with positive integer diagonal entries, and $\mathbf{b}$ is a positive vector in the kernel of $\mathcal{D} - A_G$ that satisfies $\gcd_{v \in V}(\mathbf{b}(v)) = 1$.

For each vertex $v \in V$, the processor $\mathcal{P}_v$ is given by:

$$A_v := \{v\}, \qquad Q_v := \{0, 1, \ldots, d_v - 1\}, \qquad T_v(i) := i + 1 \bmod d_v,$$

where d_v is the diagonal entry of $\mathcal{D}$ that corresponds to v. For each edge e directed from v to u, the message-passing function is given by

$$T_e(c, v) := \begin{cases} u & \text{if } c = d_v - 1; \\ \epsilon & \text{otherwise.} \end{cases}$$

Similar to sandpile networks, we can think of each processor $\mathcal{P}_v$ of this network as a locker that can store up to $d_v - 1$ chips. Once it has d_v chips, all these d_v chips in $\mathcal{P}_v$ are removed, and then $\mathcal{P}_v$ sends one chip along each of its outgoing edges to its out-neighbors. Note that the total number of chips in this network may decrease or increase, depending on the quantity $\text{outdeg}(v) - d_v$. See Figure 3.6 for an example of this process.

If $\mathcal{D}$ is the outdegree matrix of G, then $\mathcal{N}$ is the sinkless sandpile network on G. If $\mathcal{D}$ is the indegree matrix of G, then $\mathcal{N}$ is called the *row chip-firing network* [61, 1] (Note that due to a different convention for matrix indexing, this network is called the column chip-firing network in [1]).

Any arithmetical network is strongly connected if the underlying digraph G is strongly connected. The total kernel and the production matrix of this network are given by

$$K = \{\mathbf{z} \in \mathbb{Z}^V \mid \mathbf{z}(v) \text{ is divisible by } d_v \text{ for all } v \in V\};$$
$$P = A_G \mathcal{D}^{-1}.$$

Because $P(\mathcal{D}\mathbf{b}) = \mathcal{D}\mathbf{b}$ by definition, the spectral radius $\lambda(P)$ is 1 by the Perron-Frobenius theorem (Lemma 3.10(ii)). Hence an arithmetical network on a strongly connected digraph is a critical network.

There exist only finitely many arithmetical networks on a fixed strongly connected digraph [27]. For example, the bidirected cycle C_3 has ten arithmetical structures [27], namely all the permutations of these three structures:

$$\mathcal{D}_1 := \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}, \ \mathbf{b}_1 := \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}; \qquad \mathcal{D}_2 := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{bmatrix}, \ \mathbf{b}_2 := \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix};$$
$$\mathcal{D}_3 := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 5 \end{bmatrix}, \ \mathbf{b}_3 := \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}.$$

For a study of arithmetical structures on bidirected paths and cycles, we refer the reader to [27] and [18]. △

All examples presented so far are either subcritical or critical networks. In the following example we present a family of abelian networks that includes supercritical networks.

Example 3.16 (Branching rotor network). Just like for sinkless rotor networks, we assign to each $v \in V$ a cyclic total order $\{e_i^v \mid i \in \mathbb{Z}_{\mathrm{outdeg}(v)}\}$ to the outgoing edges of v. The processor $\mathcal{P}_v$ is given by

$$A_v := \{v\}, \qquad Q_v := \{e_{2i}^v \mid i \in \mathbb{Z}_{\mathrm{outdeg}(v)}\},$$
$$T_v(e_{2i}^v) := e_{2i+2}^v \quad (i \in \mathbb{Z}_{\mathrm{outdeg}(v)}).$$

(Note that $|Q_v|$ is equal to $\frac{\mathrm{outdeg}(v)}{2}$ if $\mathrm{outdeg}(v)$ is even, and is equal to $\mathrm{outdeg}(v)$ otherwise.)

For each edge e_j^v directed from v to u_j^v, the message-passing function is given by

$$T_e(e_{2i}^v, v) := \begin{cases} u_j^v & \text{if } 2i - j \in \{1,2\} \pmod{\mathrm{outdeg}(v)}; \\ \epsilon & \text{otherwise.} \end{cases}$$

Similar to sinkless rotor networks, a state of this network can be thought as a function $V \to E$ assigning a vertex v to an outgoing edge of v. When a chip/letter at vertex v is processed, the edge/state e_{2i}^v assigned to v first moves to e_{2i+1}^v and then to e_{2i+2}^v, and drops one chip at the target vertex of every visited edge. Note that branching rotor networks create two new chips for each processed chip. See Figure 3.7 for an illustration of this process.

Any branching rotor network is strongly connected if the underlying digraph G is strongly connected. The total kernel and the production matrix of this network are given by

$$K = \{\mathbf{z} \in \mathbb{Z}^V \mid \mathbf{z}(v) \text{ is divisible by } |Q_v| \text{ for all } v \in V\}; \qquad P = 2A_G D_G^{-1}.$$

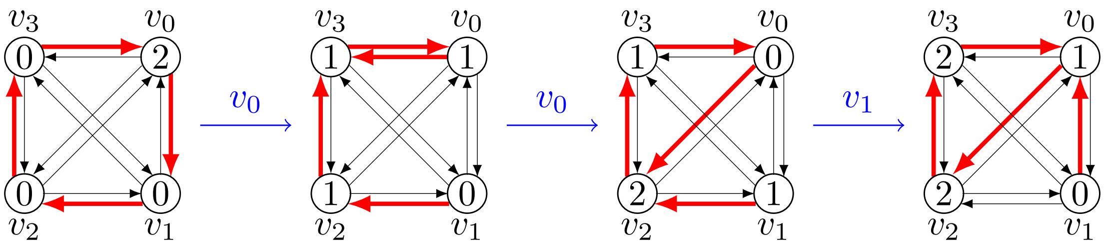

Figure 3.7: A three-step legal execution in the branching rotor network on the complete digraph with four vertices. Each vertex is assigned the counterclockwise ordering for the cyclic total order on its outgoing edges. Note that the circled number records the number of letters waiting to be processed, and the (red) thick outgoing edge records the state of the processor.

Because $\mathbf{1}A_G D_G^{-1} = \mathbf{1}$, the Perron-Frobenius theorem (Lemma 3.10(ii)) implies that $\lambda(P) = 2$, and hence this network is supercritical. $\triangle$

Example 3.17 (Toppling network [38, 15]). In a toppling network, each vertex $v \in V$ of the underlying digraph G is assigned a *threshold* $t_v \in \mathbb{N}$. For each $v \in V$, the processor $\mathcal{P}_v$ is given by

$$A_v := \{v\}, \qquad Q_v := \{0, 1, \ldots, t_v - 1\}, \qquad T_v(i) := i + 1 \bmod t_v.$$

For each edge e directed from v to u, the message-passing function is given by

$$T_e(i, v) := \begin{cases} u & \text{if } i = t_v - 1; \\ \epsilon & \text{otherwise.} \end{cases}$$

Consider now the toppling network on the bidirected cycle C_3 with $t_{v_0} = t_{v_1} = t_{v_2} =: t$. The production matrix of this network is given by:

$$P = \frac{1}{t} \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

It follows that $\lambda(P) = \frac{2}{t}$, so this network is subcritical if $t > 2$, is critical if $t = 2$, and is supercritical if $t = 1$.

Subcritical toppling networks are also known as *avalanche-finite* networks, and we refer to [42] for more discussions on this network. We remark that, on a strongly connected digraph, critical toppling networks are equal to arithmetical networks from Example 3.15. △

The following example is an instance of toppling networks that arises naturally from the representation theory.

Example 3.18 (McKay-Cartan network [6]). Let $\mathcal{G}$ be finite group, and let $\gamma : \mathcal{G} \hookrightarrow \mathtt{GL}_n(\mathbb{C})$ be a faithful representation. The underlying digraph of the McKay-Cartan network is the *McKay quiver* with vertices the complex irreducible characters $\chi_0, \ldots, \chi_k$ of $\mathcal{G}$, and with m_{ij} edges from χ_i to χ_j if

$$\chi_\gamma \chi_i = \sum_{j=0}^{k} m_{ij} \chi_j,$$

where χ_γ is the character of γ. The *McKay-Cartan* network of $(\mathcal{G}, \gamma)$ is the toppling network on the McKay quiver with threshold n for every vertex.

The production matrix of this network is equal to $\frac{1}{n}M$, where $M := (m_{i,j})_{0 \leq i,j \leq k}$ is the *extended McKay-Cartan* matrix of $(\mathcal{G}, \gamma)$. This network is strongly connected since γ is faithful [6, Proposition 5.3(c)]. Moreover, $P\mathbf{d} = \mathbf{d}$, where $\mathbf{d}(\chi_i)$ is the dimension of χ_i [6, Proposition 5.3(b)]. Hence this network is a critical network.

When γ is a faithful representation of G into the special linear group $\mathtt{SL}_n(\mathbb{C})$, the torsion group (to be defined in §4.3) of this network is isomorphic to the abelianization of $\mathcal{G}$ [6, Theorem 1.3]. △

All the examples presented so far are *unary networks*, i.e., the alphabet of each processor contains exactly one letter. In the following example we present a non-unary network.

Example 3.19 (Inverse network). For each vertex $v \in V$, fix a positive integer m_v. The processor $\mathcal{P}_v$ is given by:

$$
\begin{aligned}
&A_v := \{a_v, b_v\}, && Q_v := \mathbb{Z}_{m_v}, \\
&T_{a_v}(i) := i+1 \mod m_v, && T_{b_v}(i) := i-1 \mod m_v \quad (i \in \mathbb{Z}_{m_v}).
\end{aligned}
$$

Let c_v and d_v be two distinct letters in $\bigsqcup_{w \in \mathrm{Out}(v)} A_w$. For each $i \in \mathbb{Z}_{m_v}$, fix an element x_i from $\{c_v, d_v\}$. We define x_i^* to be

$$
x_i^* := \begin{cases} c_v & \text{if } x_i = d_v; \\ d_v & \text{if } x_i = c_v. \end{cases}
$$

The processor $\mathcal{P}_v$ operates as follows:

- Processing the letter a_v on state i produces the letter x_i; and
- Processing the letter b_v on state i produces the letter x_{i-1}^*.

For each $v \in V$, note that $t_{a_v} \circ t_{b_v} = t_{b_v} \circ t_{a_v} = \mathrm{id}$. Also note that, for all $i \in \mathbb{Z}_{m_v}$,

$$
\begin{aligned}
\mathbf{M}_{a_v b_v}(i) =& \mathbf{M}_{a_v}(i) + \mathbf{M}_{b_v}(t_{a_v}(i)) = |x_i| + |x_i^*| = |c_v| + |d_v|, \\
\mathbf{M}_{b_v a_v}(i) =& \mathbf{M}_{b_v}(i) + \mathbf{M}_{a_v}(t_{b_v}(i)) = |x_{i-1}^*| + |x_{i-1}| = |c_v| + |d_v|.
\end{aligned}
$$

This shows that inverse network is an abelian network.

The total kernel of this network is

$$
K = \{\mathbf{z} \in \mathbb{Z}^A \mid \mathbf{z}(a_v) = \mathbf{z}(b_v) \mod m_v \text{ for all } v \in V\}.
$$

Table 3.1: Example of a message-passing function of an inverse network on the digraph with one vertex and one loop. The alphabet is $\{a, b\}$ and the state space is $\mathbb{Z}_7$. The (i, α)-th entry of the table represents the letter produced when a precessor in state i processes the letter α. Note that the (i, a)-th entry is always different from the $(i + 1, b)$-th entry.

A \ Q	0	1	2	3	4	5	6
a	a	b	a	a	b	b	b
b	a	b	a	b	b	a	a

The production matrix P of any inverse network satisfies $\mathbf{1}P = \mathbf{1}$ since executing any letter in A produces exactly one new (not necessarily the same) letter. By the Perron-Frobenius theorem (Lemma 3.10(ii)) the spectral radius $\lambda(P)$ is equal to 1, and hence this network is critical. $\triangle$

CHAPTER 4

THE TORSION GROUP OF AN ABELIAN NETWORK

We start this chapter with a fundamental lemma that we call the removal lemma. We then use the removal lemma and the monoid theory from Chapter 2 to construct the torsion group for any abelian network. Finally, we show that the torsion group is equal to the critical group from [17] if the network is subcritical.

4.1 The removal lemma

Definition 4.1 (Removal of a vector from a word). For $w \in A^*$ and $\mathbf{n} \in \mathbb{N}^A$, the *removal of* $\mathbf{n}$ *from* w, denoted $w \setminus \mathbf{n}$, is the word obtained from w by deleting the first $\mathbf{n}(a)$ occurrences of a for all $a \in A$. (If a appears for less than $\mathbf{n}(a)$ times in w, then delete all occurrences of a.) $\triangle$

Recall the definition of $\dashrightarrow$, $\longrightarrow$, and legal executions from §3.2. Also recall that, for any $w \in A^*$, we denote by $|w|$ the vector in $\mathbb{N}^A$ that counts the number of occurrences of each letter in w.

The following lemma is called the *removal lemma*, as it removes some letters from a legal execution to get a shorter legal execution. A special case of this lemma when $\mathcal{N}$ is a sinkless sandpile network and $\mathbf{n}$ is the period vector (to be defined in §5.1) is proved in [12].

Lemma 4.2 (Removal lemma). *Let $\mathcal{N}$ be an abelian network, and let $\mathbf{x}.\mathbf{q}$ be a configuration of $\mathcal{N}$. Then for any $\mathbf{n} \in \mathbb{N}^A$ and any legal execution w for $\mathbf{x}.\mathbf{q}$, the word $w \setminus \mathbf{n}$ is a legal execution for $\pi_{\mathbf{n}}(\mathbf{x}.\mathbf{q})$.*

Proof. By induction on the length of the vector $\mathbf{n}$, it suffices to show that, for any $a \in A$, the word $w \setminus |a|$ is a legal execution for $\pi_a(\mathbf{x}.\mathbf{q})$.

Fix $a \in A$ throughout this proof. Let $w = a_1 \cdots a_\ell$ be the given legal execution for $\mathbf{x}.\mathbf{q}$. Let k be equal to the smallest number such that $a_k = a$ if w contains a, and equal to $\ell + 1$ if w doesn't contain a. For $i \in \{0, \ldots, \ell\}$, we write $\mathbf{x}_i.\mathbf{q}_i := \pi_{a_1 \cdots a_i}(\mathbf{x}.\mathbf{q})$ and $\mathbf{y}_i.\mathbf{p}_i := \pi_{a_1 \cdots a_i \setminus |a|}(\pi_a(\mathbf{x}.\mathbf{q}))$. We need to show that $\mathbf{y}_{i-1}(a_i) \geq 1$ for $i \in \{1, \ldots, \ell\} \setminus \{k\}$.

If $i \in \{1, \ldots, k-1\}$, then

$$\begin{aligned}
\mathbf{y}_{i-1} =& \mathbf{x} + \mathbf{M}_{aa_1 \cdots a_{i-1}}(\mathbf{q}) - |a| - \sum_{j=1}^{i-1} |a_j| \\
\geq& \mathbf{x} + \mathbf{M}_{a_1 \cdots a_{i-1}}(\mathbf{q}) - |a| - \sum_{j=1}^{i-1} |a_j| \qquad \text{(by the monotonicity property (Lemma 3.1(i)))} \\
=& \mathbf{x}_{i-1} - |a|.
\end{aligned}$$

Note that $|a|(a_i) = 0$ by the minimality of k, and also note that $\mathbf{x}_{i-1}(a_i) \geq 1$ since w is legal for $\mathbf{x}.\mathbf{q}$. Hence $\mathbf{y}_{i-1}(a_i) \geq \mathbf{x}_{i-1}(a_i) - |a|(a_i) \geq 1$.

If $i \in \{k+1, \ldots, \ell\}$, then

$$\begin{aligned}
\mathbf{y}_{i-1} =& \mathbf{x} + \mathbf{M}_{aa_1 \cdots \widehat{a_k} \cdots a_{i-1}}(\mathbf{q}) - |a| - \sum_{j \in \{1, \ldots, i\} \setminus \{k\}} |a_j| \\
=& \mathbf{x} + \mathbf{M}_{a_1 \cdots a_i}(\mathbf{q}) - \sum_{j=1}^{i-1} |a_j| \qquad \text{(by the abelian property (Lemma 3.1(ii)))} \\
=& \mathbf{x}_{i-1}.
\end{aligned}$$

Then $\mathbf{y}_{i-1}(a_i) = \mathbf{x}_{i-1}(a_i) \geq 1$ since w is legal for $\mathbf{x}.\mathbf{q}$. This completes the proof. □

Described using a diagram, the removal lemma says that

$$\begin{tikzcd} \mathbf{x.q} \arrow[r, "w"] \arrow[d, dashed, "\mathbf{n}"] & \pi_w(\mathbf{x.q}) \\ \pi_{\mathbf{n}}(\mathbf{x.q}) & \end{tikzcd} \quad \text{implies} \quad \begin{tikzcd} \mathbf{x.q} \arrow[r, "w"] \arrow[d, dashed, "\mathbf{n}"] & \pi_w(\mathbf{x.q}) \arrow[d, dashed, "\mathbf{n}\setminus|w|"] \\ \pi_{\mathbf{n}}(\mathbf{x.q}) \arrow[r, "w\setminus\mathbf{n}"] & \pi_{\max(|w|,\mathbf{n})}(\mathbf{x.q}) \end{tikzcd},$$

where $\max(\mathbf{x}, \mathbf{y})$ of two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^A$ denotes the coordinatewise maximum of $\mathbf{x}$ and $\mathbf{y}$.

Despite the apparent simplicity of the removal lemma, its consequences are very useful. One such consequence is the *least action principle.*

Recall the definition of complete execution from §3.2.

Corollary 4.3 (Least action principle [15, Lemma 4.3]). *Let $\mathcal{N}$ be an abelian network. If w is a legal execution for $\mathbf{x.q}$ and w' is a complete execution for $\mathbf{x.q}$, then $|w| \leq |w'|$.*

Proof. Since w is legal for $\mathbf{x.q}$, the removal lemma implies that $w \setminus |w'|$ is a legal execution for $\pi_{w'}(\mathbf{x.q})$. On the other hand, the only legal execution for $\pi_{w'}(\mathbf{x.q})$ is the empty word since w' is complete for $\mathbf{x.q}$. Hence $w \setminus |w'|$ is the empty word, which implies that $|w| \leq |w'|$. ☐

The second consequence of the removal lemma is the exchange lemma, presented below.

Lemma 4.4 (Exchange lemma, c.f. [13, Lemma 1.2]). *Let $\mathcal{N}$ be an abelian network. If w_1 and w_2 are two legal executions for $\mathbf{x.q}$, then there exists $w \in A^*$ such that $w_1 w$ is a legal execution for $\mathbf{x.q}$ and $|w_1| + |w| = \max(|w_1|, |w_2|)$.*

Proof. This follows from the removal lemma by taking w to be $w_2 \setminus |w_1|$. ☐

Described using a diagram, the exchange lemma says that

$$\begin{tikzcd}[row sep=small] & \mathbf{x}_2.\mathbf{q}_2 \\ \mathbf{x}_1.\mathbf{q}_1 \arrow[ur, "w_1"] \arrow[dr, "w_2"'] & \\ & \mathbf{x}_3.\mathbf{q}_3 \end{tikzcd} \quad \text{implies} \quad \begin{tikzcd}[row sep=small] & \mathbf{x}_2.\mathbf{q}_2 \arrow[dr, "w_2 \setminus |w_1|"] & \\ \mathbf{x}_1.\mathbf{q}_1 \arrow[ur, "w_1"] \arrow[dr, "w_2"'] & & \mathbf{x}_4.\mathbf{q}_4 \\ & \mathbf{x}_3.\mathbf{q}_3 \arrow[ur, "w_1 \setminus |w_2|"] & \end{tikzcd} . \tag{4.1}$$

The exchange lemma is named after a similar property of antimatroids with repetition [14]. It was proved by Björner, Lovász and Shor [13, Lemma 1.2] for sandpile networks on undirected graphs, and extended to directed graphs by Björner and Lovász [12, Proposition 1.2].

One consequence of the exchange lemma is that all abelian networks are *confluent* in the sense of Huet [46]: that is, any two legal executions w_1 and w_2 for the same configuration $\mathbf{x}.\mathbf{q}$ can be extended to longer legal executions that are equal up to a permutation of their letters (see Diagram (4.1) for an illustration). Furthermore, if the abelian network $\mathcal{N}$ is critical, then we will show that the extended execution can be taken to be of length $\max(|w_1|, |w_2|) + C$ for a constant C that depends only on the network (see Theorem 6.9).

4.2 Recurrent components

In this section we discuss recurrent components, which will be an integral ingredient in the construction of the torsion group. The reader can use the illustrations in Figure 4.1 to develop intuition when reading this section.

We start with the definition of recurrent components, which requires the notion of the trajectory digraph given below.

Definition 4.5 (Trajectory digraph). Let $\mathcal{N}$ be an abelian network. The *trajectory digraph* of $\mathcal{N}$ is the digraph with edges labeled by A given by

$$
\begin{aligned}
V &:= \{\mathbf{x}.\mathbf{q} \mid x \in \mathbb{Z}^A, \mathbf{q} \in Q\};\\
E &:= \bigsqcup_{a \in A} E_a;\\
E_a &:= \{(\mathbf{x}.\mathbf{q}, \mathbf{x}'.\mathbf{q}') \mid \mathbf{x}.\mathbf{q} \xrightarrow{a} \mathbf{x}'.\mathbf{q}'\} \quad (a \in A).
\end{aligned}
$$

△

Definition 4.6 (Quasi-legal and legal relation). Let $\mathcal{N}$ be an abelian network. Two configurations $\mathbf{x}_1.\mathbf{q}_1$ and $\mathbf{x}_2.\mathbf{q}_2$ of $\mathcal{N}$ are *quasi-legally related*, denoted $\mathbf{x}_1.\mathbf{q}_1 \dashrightarrow\!\dashleftarrow \mathbf{x}_2.\mathbf{q}_2$, if there exists $\mathbf{x}_3.\mathbf{q}_3$ such that $\mathbf{x}_1.\mathbf{q}_1 \dashrightarrow \mathbf{x}_3.\mathbf{q}_3$ and $\mathbf{x}_2.\mathbf{q}_2 \dashrightarrow \mathbf{x}_3.\mathbf{q}_3$. Two configurations $\mathbf{x}_1.\mathbf{q}_1$ and $\mathbf{x}_2.\mathbf{q}_2$ are *legally related*, denoted $\mathbf{x}_1.\mathbf{q}_1 \rightarrow\leftarrow \mathbf{x}_2.\mathbf{q}_2$, if there exists $\mathbf{x}_3.\mathbf{q}_3$ such that $\mathbf{x}_1.\mathbf{q}_1 \longrightarrow \mathbf{x}_3.\mathbf{q}_3$ and $\mathbf{x}_2.\mathbf{q}_2 \longrightarrow \mathbf{x}_3.\mathbf{q}_3$. △

The symmetry and reflexivity of these two relations follow from the definition. The transitivity of $\rightarrow\leftarrow$ follows from the exchange lemma (Lemma 4.4), because

$$
\begin{tikzcd}
\mathbf{x}_1.\mathbf{q}_1 \arrow[r] & \mathbf{x}_4.\mathbf{q}_4 \\
\mathbf{x}_2.\mathbf{q}_2 \arrow[ur, "w_1"] \arrow[dr, "w_2"'] & \\
\mathbf{x}_3.\mathbf{q}_3 \arrow[r] & \mathbf{x}_5.\mathbf{q}_5
\end{tikzcd}
\quad \text{implies} \quad
\begin{tikzcd}
\mathbf{x}_1.\mathbf{q}_1 \arrow[r] & \mathbf{x}_4.\mathbf{q}_4 \arrow[dr, "w_2\backslash|w_1|"] & \\
\mathbf{x}_2.\mathbf{q}_2 \arrow[ur, "w_1"] \arrow[dr, "w_2"'] & & \mathbf{x}_6.\mathbf{q}_6 \\
\mathbf{x}_3.\mathbf{q}_3 \arrow[r] & \mathbf{x}_5.\mathbf{q}_5 \arrow[ur, "w_1\backslash|w_2|"] &
\end{tikzcd}
. \tag{4.2}
$$

The transitivity of the quasi-legal relation is proved by an analogous diagram. Hence both relations are equivalence relations on the configurations of $\mathcal{N}$.

Definition 4.7 (Component of the trajectory digraph). Let $\mathcal{N}$ be an abelian network. A *component* of the trajectory digraph of $\mathcal{N}$ is an induced subgraph of the trajectory digraph formed by an equivalence class for the legal relation. △

See Figure 4.1 for an illustration.

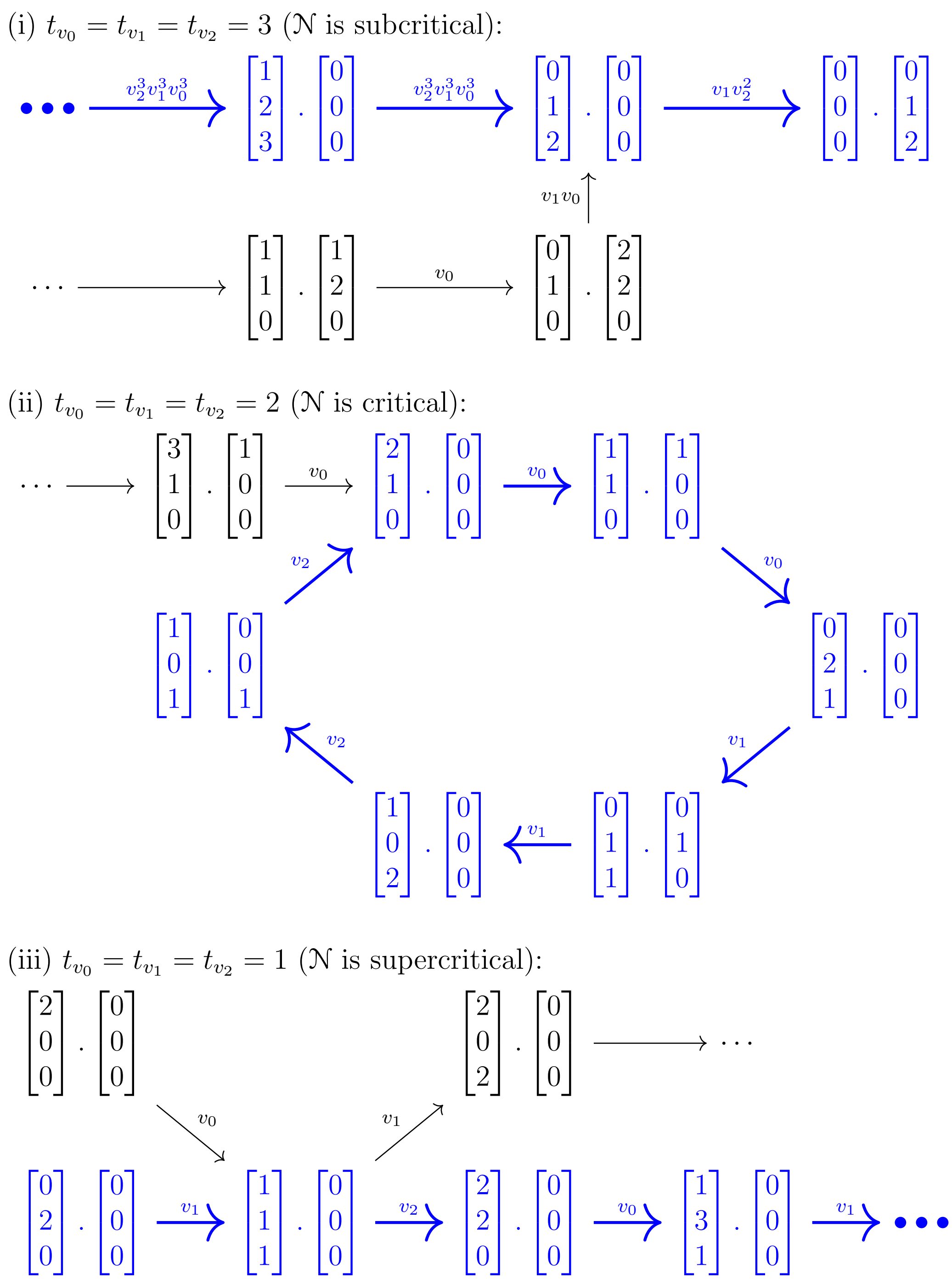

Figure 4.1: Three different toppling networks on the bidirected cycle C_3. In each case, a portion of one component of the trajectory graph is shown. The presence of a backward infinite path / cycle / forward infinite path shows that the component is recurrent.

A *forward infinite walk* in $\mathcal{N}$ is an infinite legal execution of the form $\mathbf{x}_0.\mathbf{q}_0 \xrightarrow{a_1} \mathbf{x}_1.\mathbf{q}_1 \xrightarrow{a_2} \cdots$ $(a_i \in A)$. A *backward infinite walk* is an infinite legal execution $\cdots \xrightarrow{a_{-1}} \mathbf{x}_{-1}.\mathbf{q}_{-1} \xrightarrow{a_0} \mathbf{x}_0.\mathbf{q}_0$. A *bidirectional infinite walk* is an infinite legal execution $\cdots \xrightarrow{a_0} \mathbf{x}_0.\mathbf{q}_0 \xrightarrow{a_1} \cdots$. A bidirectional infinite walk is a *cycle* if there is a positive k such that $\mathbf{x}_{i+k}.\mathbf{q}_{i+k} = \mathbf{x}_i.\mathbf{q}_i$ and $a_{i+k} = a_i$ for all $i \in \mathbb{Z}$. An *infinite walk* in $\mathcal{N}$ means either one of those three walks, i.e., a sequence $\cdots \xrightarrow{a_i} \mathbf{x}_i.\mathbf{q}_i \xrightarrow{a_{i+1}} \cdots$ indexed by I, where I is either $\mathbb{Z}_{\leq 0}$, $\mathbb{Z}_{\geq 0}$, or $\mathbb{Z}$. An *infinite path* is an infinite walk in which all $\mathbf{x}_i.\mathbf{q}_i$'s are distinct.

Definition 4.8 (Recurrent component). Let $\mathcal{N}$ be an abelian network. An infinite walk indexed by a set I is *diverse* if for all $a \in A$ the set $\{i \in I \mid a_i = a\}$ is infinite. A component of the trajectory digraph is a *recurrent component* if it contains a diverse infinite walk. $\triangle$

We denote by $\overline{\mathrm{Rec}}(\mathcal{N})$ the set of recurrent components of $\mathcal{N}$, and by $\overline{\mathbf{x}.\mathbf{q}}$ the component of the trajectory digraph that contains the configuration $\mathbf{x}.\mathbf{q}$.

Assume throughout the rest of this section that $\mathcal{N}$ is finite and locally irreducible. The first main result of this section is that, assuming recurrence, we have the quasi-legal relation implies the legal relation.

Proposition 4.9. *Let $\mathcal{N}$ be a finite and locally irreducible abelian network. If $\mathbf{x}_1.\mathbf{q}_1$ and $\mathbf{x}_2.\mathbf{q}_2$ are configurations such that $\overline{\mathbf{x}_1.\mathbf{q}_1}$ and $\overline{\mathbf{x}_2.\mathbf{q}_2}$ are recurrent components, then $\mathbf{x}_1.\mathbf{q}_1 \dashrightarrow\dashleftarrow \mathbf{x}_2.\mathbf{q}_2$ implies $\mathbf{x}_1.\mathbf{q}_1 \to\leftarrow \mathbf{x}_2.\mathbf{q}_2$.*

We remark that Proposition 4.9 for the special case of sinkless rotor networks was proved in [67, Proposition 3.7].

The second main result of this section is a trichotomy of the recurrent components of $\mathcal{N}$ that depends on the value of $\lambda(P)$.

Proposition 4.10. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected abelian network. Then the following are equivalent:*

(i) $\mathcal{N}$ *is a subcritical network;*

(ii) *All recurrent components of* $\mathcal{N}$ *contain a diverse backward infinite path; and*

(iii) *There exists a recurrent component of* $\mathcal{N}$ *that contains a diverse backward infinite path.*

Furthermore, the same statement holds if subcritical is replaced with critical (resp. supercritical) and diverse backward infinite path is replaced with diverse cycle (resp. diverse forward infinite path).

An illustration of recurrent components for each case (subcritical, critical, supercritical) is shown in Figure 4.1.

We now build towards the proof of Proposition 4.9.

Recall from §3.5 that $A_<$ denotes the set of subcritical letters of $\mathcal{N}$, and $A_\geq$ denotes the set of critical and supercritical letters of $\mathcal{N}$. We say that $\mathbf{v}, \mathbf{w} \in \mathbb{N}^A$ are *extendable vectors* of $\mathcal{N}$ if

(E1) $\operatorname{supp}(\mathbf{v}) = A_<$ and $P\mathbf{v}(a) \leq \mathbf{v}(a)$ for all $a \in A_<$;

(E2) $\operatorname{supp}(\mathbf{w}) = A_\geq$ and $P\mathbf{w}(a) \geq \mathbf{w}(a)$ for all $a \in A_\geq$; and

(E3) $\mathbf{v}$ and $\mathbf{w}$ are contained in K.

Note that extendable vectors always exist. Indeed, there exist $\mathbf{v}, \mathbf{w} \in \mathbb{N}^A$ that satisfy (E1) and (E2) by the Perron-Frobenius theorem (Lemma 3.10(vii)-(viii)).

Since the total kernel K is a subgroup of $\mathbb{Z}^A$ of finite index (by Lemma 3.7(i)), we can assume that $\mathbf{v}, \mathbf{w}$ satisfy (E3) (by taking their finite multiple if necessary).

Let $\mathbf{e} \in \mathbb{N}^A$ be a vector satisfying the conclusion of Lemma 3.4(i), i.e. for any $\mathbf{q} \in Q$ the state $t_{\mathbf{e}}(\mathbf{q})$ is locally recurrent (Note that $\mathbf{e}$ exists by Lemma 3.4(i)). The following lemma provides a method to construct diverse infinite walks.

Lemma 4.11. *Let $\mathcal{N}$ be a finite and locally irreducible abelian network. Let $\mathbf{v}, \mathbf{w}$ be extendable vectors of $\mathcal{N}$, let $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ be configurations of $\mathcal{N}$, and let $u \in A^*$ be a word such that $\mathbf{x}'.\mathbf{q}' \xrightarrow{u} \mathbf{x}.\mathbf{q}$.*

(i) *If $|u| \geq \mathbf{v} + \mathbf{e}$, then there exist $v \in A^*$ and $\mathbf{x}_{-1}, \mathbf{x}_{-2}, \ldots \in \mathbb{Z}^A$ such that $|v| = \mathbf{v}$ and the infinite execution*

$$\cdots \xrightarrow{v} \mathbf{x}_{-2}.\mathbf{q} \xrightarrow{v} \mathbf{x}_{-1}.\mathbf{q} \xrightarrow{v} \mathbf{x}.\mathbf{q}$$

is legal.

(ii) *If $|u| \geq \mathbf{w} + \mathbf{e}$, then there exist $w \in A^*$ and $\mathbf{x}_1, \mathbf{x}_2, \ldots \in \mathbb{Z}^A$ such that $|w| = \mathbf{w}$ and the infinite execution*

$$\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}_1.\mathbf{q} \xrightarrow{w} \mathbf{x}_2.\mathbf{q} \xrightarrow{w} \cdots$$

is legal.

Proof. We present only the proof of (i), as the proof of (ii) is analogous.

Write

$$v := u \setminus (|u| - \mathbf{v}); \qquad \mathbf{y}.\mathbf{p} := \pi_{|u|-|v|}(\mathbf{x}'.\mathbf{q}'); \qquad \mathbf{x}_{-i} := \mathbf{x} + i(\mathbf{y} - \mathbf{x}) \quad (i \geq 0).$$

Note that $|v| = \mathbf{v}$ since $|u| \geq \mathbf{v}$. It suffices to show that $\mathbf{x}_{-(i+1)}.\mathbf{q} \xrightarrow{v} \mathbf{x}_{-i}.\mathbf{q}$ for all $i \geq 0$.

Since $|u| - |v| \geq \mathbf{e}$ and $\mathbf{p} = t_{|u|-|v|}(\mathbf{q}')$, it follows from Lemma 3.4(i) that $\mathbf{p}$ is locally recurrent. Since $\pi_v(\mathbf{y}.\mathbf{p}) = \pi_u(\mathbf{x}'.\mathbf{q}') = \mathbf{x}.\mathbf{q}$ and $\mathbf{v} \in K$, we then have $\mathbf{q} = t_{\mathbf{v}}(\mathbf{p}) = \mathbf{p} \in \mathrm{Loc}(\mathcal{N})$ and $\mathbf{y} - \mathbf{x} = (I - P)\mathbf{v}$. Then for all $i \geq 0$,

$$\pi_v(\mathbf{x}_{-(i+1)}.\mathbf{q}) = (\mathbf{x}_{-(i+1)} - (I-P)\mathbf{v}).\mathbf{q} = \mathbf{x}_{-i}.\mathbf{q}.$$

Since $\mathbf{x}'.\mathbf{q}' \xrightarrow{u} \mathbf{x}.\mathbf{q}$ and $\pi_{|u|-|v|}(\mathbf{x}'.\mathbf{q}') = \mathbf{y}.\mathbf{q}$, the removal lemma (Lemma 4.2) implies that $\mathbf{y}.\mathbf{q} \xrightarrow{v} \mathbf{x}.\mathbf{q}$. Also note that $(\mathbf{y} - \mathbf{x})(a) = ((I-P)\mathbf{v})(a) \geq 0$ for all $a \in \mathrm{supp}(\mathbf{v})$ by (E1). It then follows from Lemma 3.3(ii) that

$$\mathbf{x}_{-(i+1)}.\mathbf{q} = (\mathbf{y} + i(\mathbf{y}-\mathbf{x})).\mathbf{q} \xrightarrow{v} (\mathbf{x} + i(\mathbf{y}-\mathbf{x})).\mathbf{q} = \mathbf{x}_{-i}.\mathbf{q},$$

for all $i \geq 0$. This completes the proof. □

As a consequence of Lemma 4.11, we show that recurrent components always exist.

Corollary 4.12. *Let $\mathcal{N}$ be a finite and locally irreducible abelian network. Then the set $\overline{\mathrm{Rec}}(\mathcal{N})$ is nonempty.*

Proof. Let $\mathbf{q}' \in Q$ and let $\mathbf{x}' := \max(\mathbf{v}, \mathbf{w}) + \mathbf{e}$, where $\mathbf{v}, \mathbf{w}$ are extendable vectors of $\mathcal{N}$. Let u be a word such that $|u| = \mathbf{x}'$. Write $\mathbf{x}.\mathbf{q} := \pi_u(\mathbf{x}'.\mathbf{q}')$, and note that $\mathbf{x}'.\mathbf{q}' \xrightarrow{u} \mathbf{x}.\mathbf{q}$ since $|u| = \mathbf{x}'$.

Since $\mathbf{v}, \mathbf{w}$ are extendable vectors, it follows from Lemma 4.11 that there exist $v, w \in A^*$ and vectors $\mathbf{x}'_i$ $(i \in \mathbb{Z} \setminus \{0\})$ such that $|v| = \mathbf{v}$, $|w| = \mathbf{w}$, and the following infinite execution

$$\cdots \xrightarrow{v} \mathbf{x}'_{-1}.\mathbf{q}' \xrightarrow{v} \mathbf{x}'.\mathbf{q}' \xrightarrow{w} \mathbf{x}'_1.\mathbf{q}' \xrightarrow{w} \cdots$$

is legal. It follows from the construction that the infinite execution above is a diverse infinite walk in $\overline{\mathbf{x}.\mathbf{q}}$. Hence $\overline{\mathbf{x}.\mathbf{q}}$ is a recurrent component, which shows that $\overline{\mathrm{Rec}}(\mathcal{N})$ is nonempty. $\square$

A *strongly diverse* infinite walk in $\mathcal{N}$ is a sequence of legal executions

$$\cdots \xrightarrow{v} \mathbf{x}_{-2}.\mathbf{q} \xrightarrow{v} \mathbf{x}_{-1}.\mathbf{q} \xrightarrow{v} \mathbf{x}_0.\mathbf{q}_0 \xrightarrow{w} \mathbf{x}_1.\mathbf{q} \xrightarrow{w} \mathbf{x}_2.\mathbf{q} \xrightarrow{w} \cdots$$

such that

(i) The state $\mathbf{q}$ is locally recurrent;

(ii) $\mathrm{supp}(|v|) = A_<$ and $P|v|(a) \leq |v|(a)$ for all $a \in A_<$; and

(iii) $\mathrm{supp}(|w|) = A_\geq$ and $P|w|(a) \geq |w|(a)$ for all $a \in A_\geq$.

Lemma 4.13. *Let $\mathcal{N}$ be a finite and locally irreducible abelian network. A component of the trajectory digraph is a recurrent component if and only if it contains a strongly diverse infinite walk.*

Proof. It suffices to prove the only if direction, as the if direction follows from the fact that a strongly diverse infinite walk is also diverse.

Let $\cdots \xrightarrow{a_i} \mathbf{x}_i.\mathbf{q}_i \xrightarrow{a_{i+1}} \cdots$ $(i \in I)$ be a diverse infinite walk in the recurrent component. Since the walk is diverse, there exist $j \in I$ and $k \geq 1$ such that $u := a_{j+1} \cdots a_{j+k}$ satisfies $|u| \geq \max(\mathbf{v}, \mathbf{w}) + \mathbf{e}$, where $\mathbf{v}, \mathbf{w}$ are extendable vectors of $\mathcal{N}$.

Write $\mathbf{x}'.\mathbf{q}' := \mathbf{x}_j.\mathbf{q}_j$ and $\mathbf{x}.\mathbf{q} := \mathbf{x}_{j+k}.\mathbf{q}_{j+k}$, and note that $\mathbf{x}'.\mathbf{q}' \xrightarrow{u} \mathbf{x}.\mathbf{q}$. Also note that we have $\mathbf{q} = t_u\mathbf{q}_j$ is locally recurrent by Lemma 3.4(i) since $|u| \geq \mathbf{e}$.

By Lemma 4.11, there exist $v, w \in A^*$ and $\mathbf{x}_i$ ($i \in \mathbb{Z} \setminus \{0\}$) such that $|v| = \mathbf{v}$, $|w| = \mathbf{w}$, and the following infinite execution

$$\cdots \xrightarrow{v} \mathbf{x}_{-2}.\mathbf{q} \xrightarrow{v} \mathbf{x}_{-1}.\mathbf{q} \xrightarrow{v} \mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}_1.\mathbf{q} \xrightarrow{w} \mathbf{x}_2.\mathbf{q} \xrightarrow{w} \cdots$$

is legal. This infinite execution is a strongly diverse infinite walk in the given recurrent component, which proves the claim. $\square$

We now present the proof of Proposition 4.9.

Proof of Proposition 4.9. By Lemma 4.13 and the transitivity of $\dashrightarrow\dashleftarrow$ and $\rightarrow\leftarrow$, we can without loss of generality assume that $\mathbf{x}_i.\mathbf{q}_i$ is contained in a strongly diverse infinite walk for $i \in \{1, 2\}$ (by taking another configuration in the recurrent component if necessary). In particular, each $\mathbf{q}_i$ is a locally recurrent state.

For $i \in \{1, 2\}$ let $\mathbf{v}_i, \mathbf{w}_i \in \mathbb{N}^A$ and $\mathbf{x}_3.\mathbf{q}_3$ be configurations such that $\mathrm{supp}(\mathbf{v}_i) = A_<$, $\mathrm{supp}(\mathbf{w}_i) = A_\geq$, and $\mathbf{x}_i.\mathbf{q}_i \overset{\mathbf{v}_i+\mathbf{w}_i}{\dashrightarrow} \mathbf{x}_3.\mathbf{q}_3$. (Note that $\mathbf{v}_i, \mathbf{w}_i$, and $\mathbf{x}_3.\mathbf{q}_3$ exist because $\mathbf{x}_1.\mathbf{q}_1 \dashrightarrow\dashleftarrow \mathbf{x}_2.\mathbf{q}_2$.) By the abelian property (Lemma 3.1(ii)) and Lemma 3.5(ii), there exist (unique) $\mathbf{x}'_i.\mathbf{q}'_i$ with $\mathbf{q}'_i \in \mathrm{Loc}(\mathcal{N})$ ($i \in \{1, 2, 3\}$) such that this diagram commutes.

$$\begin{tikzcd}
& \mathbf{x}_1.\mathbf{q}_1 \arrow[dr, dashed, "\mathbf{w}_1"] & & \\
\mathbf{x}'_1.\mathbf{q}'_1 \arrow[ur, dashed, "\mathbf{v}_2"] \arrow[dr, dashed, "\mathbf{w}_1"] & & {} \arrow[dr, dashed, "\mathbf{v}_1"] & \\
& \mathbf{x}'_3.\mathbf{q}'_3 \arrow[ur, dashed, "\mathbf{v}_2"] \arrow[dr, dashed, "\mathbf{v}_1"] & & \mathbf{x}_3.\mathbf{q}_3 \; . \\
\mathbf{x}'_2.\mathbf{q}'_2 \arrow[ur, dashed, "\mathbf{w}_2"] \arrow[dr, dashed, "\mathbf{v}_1"] & & {} \arrow[ur, dashed, "\mathbf{v}_2"] & \\
& \mathbf{x}_2.\mathbf{q}_2 \arrow[ur, dashed, "\mathbf{w}_2"] & &
\end{tikzcd}$$

For $i \in \{1,2\}$, there exist $v_i, v_i', w_i' \in A^*$, $\mathbf{y}_i, \mathbf{y}_i' \in \mathbb{Z}^A$, and $\mathbf{p}_i, \mathbf{p}_i' \in \mathrm{Loc}(\mathcal{N})$ such that (details are given after Diagram (4.3)):

$$\begin{tikzcd}
\mathbf{y}_1.\mathbf{p}_1 \arrow[rr, "v_1'"] \arrow[dr, dashed, "v_1'\backslash\mathbf{v}_2"] & & \mathbf{x}_1.\mathbf{q}_1 \arrow[rr, "w_1'"] \arrow[dr, dashed, "\mathbf{w}_1"] & & \mathbf{y}_1'.\mathbf{p}_1' \\
& \mathbf{x}_1'.\mathbf{q}_1' \arrow[ur, "v_2"] \arrow[dr, dashed, "\mathbf{w}_1"] & & {} \arrow[ur, "w_1'\backslash\mathbf{w}_1"] \arrow[dr, dashed, "\mathbf{v}_1"] & \\
& & \mathbf{x}_3'.\mathbf{q}_3' \arrow[ur, "v_2"] \arrow[dr, "v_1"] & & \mathbf{x}_3.\mathbf{q}_3\ . \\
& \mathbf{x}_2'.\mathbf{q}_2' \arrow[ur, dashed, "\mathbf{w}_2"] \arrow[dr, "v_1"] & & {} \arrow[ur, dashed, "\mathbf{v}_2"] \arrow[dr, "w_2'\backslash\mathbf{w}_2"] & \\
\mathbf{y}_2.\mathbf{p}_2 \arrow[rr, "v_2'"] \arrow[ur, dashed, "v_2'\backslash\mathbf{v}_1"] & & \mathbf{x}_2.\mathbf{q}_2 \arrow[rr, "w_2'"] \arrow[ur, dashed, "\mathbf{w}_2"] & & \mathbf{y}_2'.\mathbf{p}_2'
\end{tikzcd} \tag{4.3}$$

Indeed, let i, j be distinct elements in $\{1,2\}$. By the assumption that $\mathbf{x}_i.\mathbf{q}_i$ is contained in a strongly diverse infinite walk, we get the solid arrow $\xrightarrow{v_i'}$, where v_i' is a word such that $|v_i'| \geq \mathbf{v}_j$. Similarly, we get the solid arrow $\xrightarrow{w_i'}$, where w_i' is a word such that $|w_i'| \geq \mathbf{w}_i$. By the removal lemma (Lemma 4.2) and the assumption that $|w_i'| \geq \mathbf{w}_i$, we get the solid arrow $\xrightarrow{w_i' \backslash \mathbf{w}_i}$ in Diagram (4.3). By the abelian property, the assumption that $|v_i'| \geq \mathbf{v}_j$, and Lemma 3.5(ii), we get the dashed arrow $\overset{v_i'\backslash\mathbf{v}_j}{\dashrightarrow}$ in Diagram (4.3). Write $v_j := v_i' \setminus (|v_i'| - \mathbf{v}_j)$. Note that $|v_j| = \mathbf{v}_j$ because $|v_i'| \geq \mathbf{v}_j$, and in particular $\mathrm{supp}(|v_j|) = A_<$. By the removal lemma, we get the solid (cyan) arrow $\xrightarrow{v_j}$ in Diagram (4.3). By the removal lemma and the fact that $\mathrm{supp}(|v_j|) = A_<$ and $\mathrm{supp}(\mathbf{w}_i) = A_\geq$ are disjoint sets, we get the solid (yellow) arrow $\xrightarrow{v_j}$ in Diagram (4.3).

The conclusion of the proposition now follows from Diagram (4.3) and the transitivity of the legal relation (Diagram (4.2)). $\square$

We now build towards the proof of Proposition 4.10. We start by checking (i) implies (ii) for subcritical and supercritical case.

Lemma 4.14. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected subcritical (resp. supercritical) network. Then any strongly diverse infinite walk in $\mathcal{N}$ is a diverse backward (resp. forward) infinite path.*

Proof. We present only the proof of the subcritical case, as the proof of the supercritical case is analogous.

Since $\mathcal{N}$ is subcritical, a strongly diverse infinite walk of $\mathcal{N}$ is of the form

$$\cdots \xrightarrow{v} \mathbf{x}_{-2}.\mathbf{q} \xrightarrow{v} \mathbf{x}_{-1}.\mathbf{q} \xrightarrow{v} \mathbf{x}.\mathbf{q},$$

where v is a word such that $\operatorname{supp}(|v|) = A$. Note that the infinite execution above is a diverse backward infinite walk. Hence it suffices to show that this infinite walk is a path.

Suppose to the contrary that this infinite walk is not a path. Then there exists a configuration $\mathbf{x}'.\mathbf{q}'$ and a nonempty word w such that the execution $\mathbf{x}'.\mathbf{q}' \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$ is legal and is a subsequence of the infinite walk above. By Lemma 3.9, we then have:

$$P|w| = \mathbf{x}' + |w| - \mathbf{x}' = |w|.$$

The Perron-Frobenius theorem (Lemma 3.10(iii)) then implies that $\lambda(P) = 1$, contradicting the assumption that $\mathcal{N}$ is subcritical. This proves the claim. $\square$

We will use the following version of Dickson's lemma to check (i) implies (ii) for critical case. A sequence of vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots$ in $\mathbb{Z}^A$ has a *lower bound* if there exists $\mathbf{x} \in \mathbb{Z}^A$ such that $\mathbf{x}_i \geq \mathbf{x}$ for all $i \geq 1$.

Lemma 4.15 ([32, Dickson's lemma]). *Let $\mathbf{x}_1, \mathbf{x}_2, \ldots$ be a sequence of vectors in $\mathbb{Z}^A$ that has a lower bound. Then there exist integers $j, k \geq 1$ such that $\mathbf{x}_j \leq \mathbf{x}_{j+k}$.* $\square$

Denote by $\mathbf{0}$ the vector in $\mathbb{Z}^A$ with all entries being equal to 0.

Lemma 4.16. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected critical network. Then any strongly diverse infinite walk in $\mathcal{N}$ is a diverse cycle.*

Proof. Since $\mathcal{N}$ is critical, a strongly diverse infinite walk of $\mathcal{N}$ is of the form

$$\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}_1.\mathbf{q} \xrightarrow{w} \mathbf{x}_2.\mathbf{q} \xrightarrow{w} \cdots$$

where w is a word such that $\operatorname{supp}(|w|) = A$. Hence it suffices to show $\mathbf{x}_1 = \mathbf{x}$.

By Lemma 3.3(iii), the sequence $\mathbf{x}_0, \mathbf{x}_1, \ldots$ is lower bounded by the vector $\mathbf{x} \in \mathbb{Z}^A$ given by $\mathbf{x}(a) := \min\{\mathbf{x}_0(a), 0\}$ $(a \in A)$. By Dickson's lemma, there exist integers $j, k \geq 1$ such that $\mathbf{x}_j \leq \mathbf{x}_{j+k}$.

Since $\mathbf{x}_j.\mathbf{q}_j \xrightarrow{w^k} \mathbf{x}_{j+k}.\mathbf{q}_{j+k}$ and $\mathbf{q}_{j+k} = \mathbf{q}_j$, we have by Lemma 3.9 that

$$(P - I)|w| = \frac{\mathbf{x}_{j+k} - \mathbf{x}_j}{k} \geq \mathbf{0}.$$

Since $\mathcal{N}$ is strongly connected and critical, it follows from the Perron-Frobenius theorem (Lemma 3.10(iii)) that $(P - I)|w| = \mathbf{0}$. This implies that

$$\mathbf{x}_1 - \mathbf{x} = (P - I)|w| = \mathbf{0},$$

as desired. □

We now present the proof of Proposition 4.10.

Proof of Proposition 4.10. (i) implies (ii): This follows from Lemma 4.13, Lemma 4.14, and Lemma 4.16.

(ii) implies (iii) is straightforward.

(iii) implies (i): We present only the proof of the subcritical case, as the proof of the other two cases are analogous.

By (iii), there exists a diverse infinite path in $\mathcal{N}$ of the form

$$\cdots \xrightarrow{v_3} \mathbf{x}_{-2}.\mathbf{q} \xrightarrow{v_2} \mathbf{x}_{-1}.\mathbf{q} \xrightarrow{v_1} \mathbf{x}.\mathbf{q},$$

where $v_1, v_2 \ldots$ are words such that $\mathrm{supp}(|v_i|) = A$. Note that $\mathbf{x}_i \neq \mathbf{x}_j$ for distinct i and j since the infinite walk above is a path.

Since $\mathbf{x}_{-(i+1)}.\mathbf{q} \xrightarrow{v_{i+1}} \mathbf{x}_{-i}.\mathbf{q}$ and $\mathrm{supp}(|v_{i+1}|) = A$ for any $i \geq 0$, we have by Lemma 3.3(iii) that $\mathbf{x}_{-i}$ is a nonnegative vector for any $i \geq 0$. By Dickson's lemma, there exist integers $j, k \geq 1$ such that $\mathbf{x}_{-j} \leq \mathbf{x}_{-(j+k)}$.

Write $v := v_k v_{k-1} \ldots v_{j+1}$. Now note that

$$(I - P)|v| = \mathbf{x}_{-(j+k)} - \mathbf{x}_{-j} \geq \mathbf{0},$$

where the first equality is due to Lemma 3.9. Also note that $(I-P)|v| = \mathbf{x}_{-(j+k)} - \mathbf{x}_{-j}$ is not equal to $\mathbf{0}$ since $\mathbf{x}_{-(j+k)} \neq \mathbf{x}_{-j}$ by assumption. Since $\mathcal{N}$ is strongly connected, it then follows from the Perron-Frobenius theorem (Lemma 3.10(iii)) that $\lambda(P)$ is strictly less than 1, as desired. $\square$

4.3 Construction of the torsion group

In this section we define the torsion group for any abelian network by building on results from §4.2. The reader can use the networks from Example 3.17 to develop intuition when reading this section.

Definition 4.17 (Shift monoid). Let $\mathcal{N}$ be an abelian network. The monoid $\mathbb{N}^A$

acts on $\overline{\mathrm{Rec}}(\mathcal{N})$ by

$$\phi : \mathbb{N}^A \to \mathrm{End}(\overline{\mathrm{Rec}}(\mathcal{N}))$$
$$\phi(\mathbf{n})(\overline{\mathbf{x}.\mathbf{q}}) := \overline{(\mathbf{x}+\mathbf{n}).\mathbf{q}}.$$

The *shift monoid* is the monoid $\mathcal{M}(\mathcal{N}) := \phi(\mathbb{N}^A)$. △

It follows from Lemma 3.3(ii) that $\overline{(\mathbf{x}+\mathbf{n}).\mathbf{q}}$ does not depend on the choice of $\mathbf{x}.\mathbf{q}$, and is a recurrent component if $\overline{\mathbf{x}.\mathbf{q}}$ is recurrent. Hence the monoid action in Definition 4.17 is well-defined.

Note that $\mathcal{M}(\mathcal{N})$ is generated by the set $\{\phi(|a|) \mid a \in A\}$, and hence is a finitely generated commutative monoid. We denote by $\mathcal{K}(\mathcal{N})$ the Grothendieck group (see §2.1) of $\mathcal{M}$. We remark that $\mathcal{M}(\mathcal{N})$, $\mathcal{K}(\mathcal{N})$, and $\overline{\mathrm{Rec}}(\mathcal{N})$ can be infinite; see Example 4.22(ii).

Definition 4.18 (Torsion group). Let $\mathcal{N}$ be an abelian network. The *torsion group* of $\mathcal{N}$ is

$$\mathrm{Tor}(\mathcal{N}) := \tau(\mathcal{K}(\mathcal{N})),$$

the torsion subgroup of the Grothendieck group of $\mathcal{M}(\mathcal{N})$. △

Definition 4.19 (Invertible recurrent component). Let $\mathcal{N}$ be an abelian network. A recurrent component $\overline{\mathbf{x}.\mathbf{q}}$ is *invertible* if, for any $g \in \mathrm{Tor}(\mathcal{N})$ and any $\mathbf{n}, \mathbf{n}' \in \mathbb{N}^A$ such that $g = \overline{(\phi(\mathbf{n}), \phi(\mathbf{n}'))}$, there exists $\overline{\mathbf{x}'.\mathbf{q}'} \in \overline{\mathrm{Rec}}(\mathcal{N})$ such that

$$\overline{(\mathbf{x}+\mathbf{n}).\mathbf{q}} = \overline{(\mathbf{x}'+\mathbf{n}').\mathbf{q}'}.$$

We denote by $\overline{\mathrm{Rec}}(\mathcal{N})^\times$ the set of invertible recurrent components of $\mathcal{N}$. △

Note that not all recurrent components are invertible; see Example 4.22(ii).

Assume throughout the rest of this section that $\mathcal{N}$ is a finite and locally irreducible abelian network, unless stated otherwise.

Definition 4.20 (Action of $\mathrm{Tor}(\mathcal{N})$ on $\overline{\mathrm{Rec}}(\mathcal{N})^{\times}$). Let $\mathcal{N}$ be a finite and locally irreducible abelian network. The group $\mathrm{Tor}(\mathcal{N})$ acts on $\overline{\mathrm{Rec}}(\mathcal{N})^{\times}$ by

$$\mathrm{Tor}(\mathcal{N}) \times \overline{\mathrm{Rec}}(\mathcal{N})^{\times} \to \overline{\mathrm{Rec}}(\mathcal{N})^{\times}$$
$$(g, \overline{\mathbf{x}.\mathbf{q}}) \mapsto \overline{\mathbf{x}'.\mathbf{q}'},$$

where $\overline{\mathbf{x}'.\mathbf{q}'}$ is as in Definition 4.19. △

We will show later in Lemma 4.23(iii) that this group action is well-defined. Note that the action of $\mathrm{Tor}(\mathcal{N})$ is not defined for recurrent components that are not invertible.

We now state the main result of this section. Recall the definition of the total kernel K (Definition 3.6) and the production matrix P (Definition 3.8). Recall that the action of a monoid $\mathcal{M}$ on a set X is *free* if, for any $x \in X$ and $m, m' \in \mathcal{M}$, we have $mx = m'x$ implies that $m = m'$. The action of $\mathcal{M}$ on X is *transitive* if X is nonempty and for any $x, x' \in X$ there exists $m \in \mathcal{M}$ such that $x' = mx$.

Theorem 4.21. *Let $\mathcal{N}$ be a finite and locally irreducible abelian network. Then*

(i) $\overline{\mathrm{Rec}}(\mathcal{N})^{\times}$ *is nonempty.*

(ii) $\mathrm{Tor}(\mathcal{N})$ *is a finite abelian group that acts freely on* $\overline{\mathrm{Rec}}(\mathcal{N})^{\times}$.

(iii) *The map* $\phi : \mathbb{N}^A \to \mathrm{End}(\mathrm{Rec}(\mathcal{N}))$ *induces an isomorphism of abelian groups*

$$\mathcal{K}(\mathcal{N}) \simeq \mathbb{Z}^A/(I - P)K.$$

We remark that Theorem 1.1, stated in the introduction, is a direct corollary of Theorem 4.21.

Note that the action of the torsion group on $\overline{\mathrm{Rec}}(\mathcal{N})^\times$ is in general not transitive; see Example 4.22(ii). The torsion group is a generalization of the critical group for halting networks as defined in [17]. We will discuss this in more detail in §4.4.

Example 4.22. Consider the toppling network $\mathcal{N}_t$ (Example 3.17) on the bidirected cycle C_3 with threshold $t_{v_0} = t_{v_1} = t_{v_2} =: t$.

(i) If $t = 3$ (note that $\mathcal{N}_3$ is subcritical), then

$$\mathrm{Tor}(\mathcal{N}_3) = \mathbb{Z}^V \Big/ \left\langle \begin{bmatrix} 3 \\ -1 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 3 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \\ 3 \end{bmatrix} \right\rangle_{\mathbb{Z}} = \mathbb{Z}_4 \oplus \mathbb{Z}_4.$$

$\mathcal{N}_3$ has sixteen recurrent components, namely all permutations of these five:

$$\left\{ \overline{\mathbf{x}.\mathbf{q}} \;\middle|\; \mathbf{x} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \mathbf{q} \in \left\{ \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix} \right\} \right\}.$$

All sixteen recurrent components of $\mathcal{N}_3$ are invertible, and the action of $\mathrm{Tor}(\mathcal{N}_3)$ on $\overline{\mathrm{Rec}}(\mathcal{N}_3)^\times$ is free and transitive.

(ii) If $t = 2$ (note that $\mathcal{N}_2$ is critical), then:

$$\begin{aligned}\mathrm{Tor}(\mathcal{N}_2) =& \tau \left(\mathbb{Z}^V \Big/ \left\langle \begin{bmatrix} 2 \\ -1 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 2 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \\ 2 \end{bmatrix} \right\rangle_{\mathbb{Z}} \right) \\ =& \tau(\mathbb{Z}_3 \oplus \mathbb{Z}) = \mathbb{Z}_3.\end{aligned}$$

The recurrent components of $\mathcal{N}_2$ are given by

$$\overline{\mathrm{Rec}}(\mathcal{N}_2) = \bigsqcup_{m \geq 3} \overline{\mathrm{Rec}}(\mathcal{N}_2, m),$$

where

$$\overline{\mathrm{Rec}}(\mathcal{N}_2, 3) = \left\{ \overline{\mathbf{x}.\mathbf{q}} \;\middle|\; \mathbf{q} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},\ \mathbf{x} \in \left\{ \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix} \right\} \right\},$$

and, for $m \geq 4$,

$$\overline{\mathrm{Rec}}(\mathcal{N}_2, m) = \left\{ \overline{\mathbf{x}.\mathbf{q}} \;\middle|\; \mathbf{q} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},\ \mathbf{x} \in \left\{ \begin{bmatrix} 0 \\ 1 \\ m-1 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \\ m-2 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ m-2 \end{bmatrix} \right\} \right\}.$$

The invertible recurrent components of $\mathcal{N}_2$ are given by:

$$\overline{\mathrm{Rec}}(\mathcal{N}_2)^{\times} = \bigsqcup_{m \geq 4} \overline{\mathrm{Rec}}(\mathcal{N}_2, m).$$

In particular, the two recurrent components in $\overline{\mathrm{Rec}}(\mathcal{N}_2, 3)$ are not invertible, and hence the torsion group does not act on them.

Note that the action of $\mathrm{Tor}(\mathcal{N}_2)$ on $\overline{\mathrm{Rec}}(\mathcal{N}_2)^{\times}$ is free but not transitive, as each $\overline{\mathrm{Rec}}(\mathcal{N}_2, m)$ for $m \geq 4$ is an orbit of this action.

(iii) If $t = 1$ (note that $\mathcal{N}_1$ is supercritical), then

$$\mathrm{Tor}(\mathcal{N}_1) = \mathbb{Z}^V \Big/ \left\langle \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} \right\rangle_{\mathbb{Z}} = \mathbb{Z}_2 \oplus \mathbb{Z}_2.$$

$\mathcal{N}_1$ has four recurrent components:

$$\left\{ \overline{\mathbf{x}.\mathbf{q}} \;\middle|\; \mathbf{q} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},\ \mathbf{x} \in \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\} \right\}.$$

All four recurrent components of $\mathcal{N}_1$ are invertible, and the action of $\mathrm{Tor}(\mathcal{N}_1)$ on $\overline{\mathrm{Rec}}(\mathcal{N}_1)^{\times}$ is free and transitive. $\triangle$

Our strategy of proving Theorem 4.21 is to apply Proposition 2.5 to the setting of Theorem 4.21. In order to do so, we need to check that the action of $\mathcal{M}(\mathcal{N})$ on $\overline{\mathrm{Rec}}(\mathcal{N})$ satisfies the conditions in Proposition 2.5, and that requires the following technical lemma.

Recall the definition of injective action from Definition 2.1.

Lemma 4.23. *Let $\mathcal{N}$ be a finite and locally irreducible abelian network. Then*

(i) *For any $\mathbf{n}, \mathbf{n}' \in \mathbb{N}^A$, we have $\phi(\mathbf{n}) = \phi(\mathbf{n}')$ if and only if $\mathbf{n} - \mathbf{n}' \in (I-P)K$;*

(ii) *The action of $\mathcal{M}(\mathcal{N})$ on $\overline{\mathrm{Rec}}(\mathcal{N})$ is free and injective; and*

(iii) *The action of $\mathrm{Tor}(\mathcal{N})$ on $\overline{\mathrm{Rec}}(\mathcal{N})^\times$ in Definition 4.20 is well defined.*

Proof. Let $\mathbf{x}.\mathbf{q}$ be any configuration such that $\overline{\mathbf{x}.\mathbf{q}}$ is recurrent. For any $\mathbf{n}, \mathbf{n}' \in \mathbb{N}^A$,

$$\begin{aligned} &(\mathbf{x}+\mathbf{n}).\mathbf{q} \to\leftarrow (\mathbf{x}+\mathbf{n}').\mathbf{q} \\ \iff\quad &(\mathbf{x}+\mathbf{n}).\mathbf{q} \dashrightarrow\dashleftarrow (\mathbf{x}+\mathbf{n}').\mathbf{q} \qquad \text{(by Proposition 4.9)} \\ \iff\quad &\mathbf{n}-\mathbf{n}' \in (I-P)K \qquad \text{(by Lemma 3.9).} \end{aligned} \tag{4.4}$$

Since the choice of $\mathbf{x}.\mathbf{q}$ is arbitrary, we then conclude that $\phi(\mathbf{n}) = \phi(\mathbf{n}')$ if and only if $\mathbf{n} - \mathbf{n}' \in (I-P)K$. This proves part (i).

Let $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ be any configurations such that $\overline{\mathbf{x}.\mathbf{q}}$ and $\overline{\mathbf{x}'.\mathbf{q}'}$ are recurrent. For any $\mathbf{n} \in \mathbb{N}^A$,

$$\begin{aligned} &(\mathbf{x}+\mathbf{n}).\mathbf{q} \to\leftarrow (\mathbf{x}'+\mathbf{n}).\mathbf{q}' \\ \Longrightarrow\quad &(\mathbf{x}+\mathbf{n}).\mathbf{q} \dashrightarrow\dashleftarrow (\mathbf{x}'+\mathbf{n}).\mathbf{q}' \\ \Longrightarrow\quad &\mathbf{x}.\mathbf{q} \dashrightarrow\dashleftarrow \mathbf{x}'.\mathbf{q}' \qquad \text{(by Lemma 3.3(i))} \\ \Longrightarrow\quad &\mathbf{x}.\mathbf{q} \to\leftarrow \mathbf{x}'.\mathbf{q}' \qquad \text{(by Proposition 4.9).} \end{aligned}$$

Hence the action of $\mathcal{M}(\mathcal{N})$ on $\overline{\mathrm{Rec}}(\mathcal{N})$ is injective. For any $\mathbf{n}, \mathbf{n}' \in \mathbb{N}^A$,

$$\begin{aligned}
&(\mathbf{x}+\mathbf{n}).\mathbf{q} \to\leftarrow (\mathbf{x}+\mathbf{n}').\mathbf{q} \\
\implies \quad & \mathbf{n}-\mathbf{n}' \in (I-P)K \qquad \text{(by (4.4))} \\
\implies \quad & (\mathbf{x}'+\mathbf{n}).\mathbf{q}' \to\leftarrow (\mathbf{x}'+\mathbf{n}').\mathbf{q}' \qquad \text{(by (4.4))}.
\end{aligned}$$

Since the choice of $\mathbf{x}'.\mathbf{q}'$ is arbitrary, we then conclude that $\phi(\mathbf{n})(\mathbf{x}.\mathbf{q}) = \phi(\mathbf{n}')(\mathbf{x}.\mathbf{q})$ implies that $\phi(\mathbf{n}) = \phi(\mathbf{n}')$. Hence the the action of $\mathcal{M}(\mathcal{N})$ on $\overline{\mathrm{Rec}}(\mathcal{N})$ is free. This proves part (ii).

Since $\mathcal{M}(\mathcal{N})$ acts on $\overline{\mathrm{Rec}}(\mathcal{N})$ injectively by part (ii), it follows from Lemma 2.3 that the group action in Definition 4.20 is well-defined. This proves part (iii). □

We now present the proof of Theorem 4.21.

Proof of Theorem 4.21. Note that action of $\mathcal{M}(\mathcal{N})$ on $\overline{\mathrm{Rec}}(\mathcal{N})$ is free and injective (by Lemma 4.23(ii)), and that $\overline{\mathrm{Rec}}(\mathcal{N})$ is a nonempty set (by Corollary 4.12). Part (i) and (ii) now follow directly from Proposition 2.5.

For part (iii), note that $\mathbb{Z}^A$ is the Grothendieck group of $\mathbb{N}^A$ and $\mathcal{K}(\mathcal{N})$ is the Grothendieck group of $\mathcal{M}(\mathcal{N})$. Also note that $\phi : \mathbb{N}^A \to \mathcal{M}(\mathcal{N})$ is a surjective monoid homomorphism. By the universal property of the Grothendieck group, the map ϕ induces a surjective group homomorphism $\phi : \mathbb{Z}^A \to \mathcal{K}(\mathcal{N})$. Also note that

$$\ker(\phi) = \{\mathbf{z} \in \mathbb{Z}^A \mid \phi(\mathbf{z}^+) = \phi(\mathbf{z}^-)\},$$

where $\mathbf{z}^+$ and $\mathbf{z}^-$ are the positive part and the negative part of $\mathbf{z}$, respectively. The claim now follows from Lemma 4.23(i). □

4.4 Relations to the critical group in the halting case

Consider a finite, locally irreducible, and subcritical abelian network $\mathcal{S}$. In this section we show that the torsion group of $\mathcal{S}$ is isomorphic to the critical group defined in [17].

We start by quoting a useful theorem from [17]. A configuration $\mathbf{x}.\mathbf{q}$ is *stable* if $\mathbf{x}(a) \leq 0$ for all $a \in A$. A configuration $\mathbf{x}.\mathbf{q}$ halts if there exists a stable configuration $\mathbf{x}'.\mathbf{q}'$ such that $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$.

Theorem 4.24 ([16, Theorem 5.6]). *Let $\mathcal{S}$ be a finite, locally irreducible, and subcritical abelian network. Then every configuration $\mathbf{x}.\mathbf{q}$ in $\mathcal{S}$ is a halting configuration.* □

Lemma 4.25. *Let $\mathcal{S}$ be a finite, locally irreducible, and subcritical abelian network. Then every component of the trajectory digraph contains a unique stable configuration.*

Proof. Let $\mathcal{C}$ be an arbitrary component of the trajectory digraph. By Theorem 4.24, there exists a stable configuration $\mathbf{x}.\mathbf{q}$ in $\mathcal{C}$.

We now prove that $\mathbf{x}.\mathbf{q}$ is unique. Let $\mathbf{x}'.\mathbf{q}'$ be another stable configuration in $\mathcal{C}$. Then there exists $\mathbf{y}.\mathbf{p}$ such that $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{y}.\mathbf{p}$ and $\mathbf{x}'.\mathbf{q}' \longrightarrow \mathbf{y}.\mathbf{p}$. Since $\mathbf{x}(a) \leq 0$ for all $a \in A$, it is necessary that $\mathbf{x}.\mathbf{q} = \mathbf{y}.\mathbf{p}$. By symmetry $\mathbf{x}'.\mathbf{q}' = \mathbf{y}.\mathbf{p}$, and hence $\mathbf{x}.\mathbf{q} = \mathbf{x}'.\mathbf{q}'$. □

We define the *stabilization* $\mathsf{ST}(\mathcal{C})$ of a component $\mathcal{C}$ to be the unique stable configuration in $\mathcal{C}$. Let $\mathcal{Q}$ be the set:

$$\mathcal{Q} := \{\mathcal{C} \mid \mathsf{ST}(\mathcal{C}) = \mathbf{0}.\mathbf{q} \text{ for some } \mathbf{q} \in Q\}.$$

The set $\mathcal{Q}$ is in one-to-one correspondence with the state space Q via $\overline{\mathbf{0}.\mathbf{q}} \mapsto \mathbf{q}$, and in particular $\mathcal{Q}$ is finite.

The monoid $\mathbb{N}^A$ acts on $\mathcal{Q}$ by:

$$\Phi : \mathbb{N}^A \to \mathrm{End}(\mathcal{Q})$$
$$\Phi(\mathbf{n})(\overline{\mathbf{0}.\mathbf{q}}) := \overline{\mathbf{n}.\mathbf{q}}.$$

Note that $\mathsf{ST}(\overline{\mathbf{n}.\mathbf{q}}) = \mathbf{0}.\mathbf{q}'$ for some $\mathbf{q}' \in Q$ since $\mathbf{n} \geq \mathbf{0}$, and hence $\overline{\mathbf{n}.\mathbf{q}}$ is contained in $\mathcal{Q}$.

The *global monoid* in the sense of [17] is the monoid $\mathcal{F}(\mathcal{S}) := \Phi(\mathbb{N}^A)$. Note that $\mathcal{F}(\mathcal{S})$ is a finite commutative monoid as $\mathcal{Q}$ is finite.

Let $e \in \mathcal{F}(\mathcal{S})$ be the minimal idempotent of $\mathcal{F}(\mathcal{S})$ (see Definition 2.6). The *critical group* of $\mathcal{N}$ in the sense of [17] is the group $e\mathcal{F}(\mathcal{S})$.

Definition 4.26 (Recurrent state). Let $\mathcal{S}$ be a finite and locally irreducible subcritical network. An element of $\mathcal{Q}$ is *recurrent* in the sense of [17] if it is contained in $e\mathcal{Q}$. A state $\mathbf{q} \in Q$ is *recurrent* if its corresponding component in $\mathcal{Q}$ is a recurrent component. $\triangle$

We now explain how these objects from [17] fit into our work. Recall that $\overline{\mathrm{Rec}}(\mathcal{S})$ is the set of recurrent components of $\mathcal{S}$ (see Definition 4.8).

Lemma 4.27. *Let $\mathcal{S}$ be a finite, locally irreducible, and subcritical abelian network. Then $\overline{\mathrm{Rec}}(\mathcal{S})$ is a closed subset of $\mathcal{Q}$ under the action of $\mathcal{F}(\mathcal{S})$.*

Proof. We first show that the set $\overline{\mathrm{Rec}}(\mathcal{S})$ is a subset of $\mathcal{Q}$. Let $\mathcal{C}$ be any recurrent component of $\mathcal{S}$, and let $\mathbf{x}.\mathbf{q} := \mathsf{ST}(\mathcal{C})$. Since $\mathcal{S}$ is subcritical and $\mathcal{C}$ is recurrent, by Lemma 4.13 there exist a configuration $\mathbf{x}'.\mathbf{q}'$ and $w \in A^*$ such that $\mathbf{x}'.\mathbf{q}' \xrightarrow{w} \mathbf{x}.\mathbf{q}$

and $|w| \geq \mathbf{1}$. By Lemma 3.3(iii) and the fact that $\mathbf{x}.\mathbf{q}$ is stable, we conclude that that $\mathbf{x} = \mathbf{0}$. This then implies that $\mathcal{C}$ is in $\mathcal{Q}$.

Let $\mathbf{n}$ be any nonnegative vector and let $\overline{\mathbf{x}.\mathbf{q}}$ be any recurrent component. It follows from Lemma 3.3(ii) and the definition of recurrence that $\overline{(\mathbf{x}+\mathbf{n}).\mathbf{q}}$ is a recurrent component. This shows that $\overline{\mathrm{Rec}}(\mathcal{S})$ is closed under the action of $\mathcal{F}(\mathcal{S})$. $\square$

Let $\eta : \mathcal{F}(\mathcal{S}) \to \mathrm{End}(\overline{\mathrm{Rec}}(\mathcal{S}))$ be the monoid homomorphism induced by the action of $\mathcal{F}(\mathcal{S})$ on $\overline{\mathrm{Rec}}(\mathcal{S})$. Note that the shift monoid $\mathcal{M}(\mathcal{S})$ from Definition 4.17 is the image of the global monoid $\mathcal{F}(\mathcal{S})$ under the map η. We denote by ϵ the identity of element of $\mathcal{M}(\mathcal{S})$.

Recall that the torsion group $\mathrm{Tor}(\mathcal{S})$ is the torsion subgroup of the Grothendieck group of $\mathcal{M}(\mathcal{S})$, and $\mathrm{Tor}(\mathcal{S})$ acts on the set of invertible recurrent components $\overline{\mathrm{Rec}}(\mathcal{S})^{\times}$ (see Definition 4.19).

We now state a theorem which shows that, for a subcritical network, the construction in [17] and our construction give rise to the same group.

Theorem 4.28. *Let $\mathcal{S}$ be a finite, locally irreducible, and subcritical abelian network. Then*

(i) $e\mathcal{F}(\mathcal{S}) \simeq \mathrm{Tor}(\mathcal{S})$ *by the map* $F : e\mathcal{F}(\mathcal{S}) \to \mathrm{Tor}(\mathcal{S})$ *defined by* $em \mapsto \overline{(\eta(em), \epsilon)}$.

(ii) $e\mathcal{Q} = \overline{\mathrm{Rec}}(\mathcal{S}) = \overline{\mathrm{Rec}}(\mathcal{S})^{\times}$.

(iii) *The isomorphism* $F : e\mathcal{F}(\mathcal{S}) \to \mathrm{Tor}(\mathcal{S})$ *preserves the action of* $e\mathcal{F}(\mathcal{S})$ *and* $\mathrm{Tor}(\mathcal{S})$ *on* $e\mathcal{Q} = \overline{\mathrm{Rec}}(\mathcal{S})^{\times}$.

Proof. We first check that the assumptions in Proposition 2.8 are satisfied. The action of $\mathcal{F}(\mathcal{S})$ on $\mathcal{Q}$ is faithful by definition. We now show that the action of $\mathcal{F}(\mathcal{S})$ on $\mathcal{Q}$ is irreducible. Let $\overline{\mathbf{0}.\mathbf{q}}$ and $\overline{\mathbf{0}.\mathbf{q}'}$ be any two elements of $\mathcal{Q}$. Since $\mathcal{S}$ is locally irreducible, there exist $w, w' \in A^*$ such that $t_w \mathbf{q} = t_{w'} \mathbf{q}'$. Then there exist $\mathbf{n}, \mathbf{n}', \mathbf{m} \in \mathbb{N}^A$ such that $\mathbf{n}.\mathbf{q} \xrightarrow{w} \mathbf{m}.t_w(\mathbf{q})$ and $\mathbf{n}'.\mathbf{q}' \xrightarrow{w'} \mathbf{m}.t_{w'}(\mathbf{q}')$. These two facts imply that $\Phi(\mathbf{n})(\overline{\mathbf{0}.\mathbf{q}}) = \Phi(\mathbf{n}')(\overline{\mathbf{0}.\mathbf{q}'})$, which proves irreducibility. Also note that the set $\mathcal{Q}$ is nonempty since Q is nonempty by the definition of abelian networks.

Note that $\overline{\mathrm{Rec}}(\mathcal{S})$ is nonempty (by Corollary 4.12), is a closed subset of $\mathcal{Q}$ (by Lemma 4.27), and the action of $\mathcal{F}(\mathcal{S})$ on it is injective (by Lemma 4.23(ii)). The theorem now follows from Proposition 2.8. $\square$

CHAPTER 5

CRITICAL NETWORKS: RECURRENCE

In this chapter we study critical networks in more detail, with a focus on their recurrent configurations and torsion group. Examples of critical networks include sinkless rotor networks (Example 3.11), sinkless sandpile networks (Example 3.12), sinkless height-arrow networks (Example 3.13), arithmetical networks (Example 3.15), and inverse networks (Example 3.19).

5.1 Recurrent configurations and the burning test

In this section we define the notion of recurrence for configurations of a critical network, and we outline a test to check for the recurrence of a configuration.

We assume throughout this section that $\mathcal{N}$ is a finite, locally irreducible, and strongly connected critical network unless stated otherwise.

Integral to our study of critical networks is the notion of period vector, defined as follows.

Denote by $\mathcal{E}$ the (right) eigenspace of $\lambda(P)$ of the production matrix P of $\mathcal{N}$. By the Perron-Frobenius theorem (Lemma 3.10(vi)), the vector space $\mathcal{E}$ is spanned by a positive integer vector. Since the total kernel K is a subgroup of $\mathbb{Z}^A$ of finite index (Lemma 3.7(i)), the set $\mathcal{E} \cap K$ is equal to the $\mathbb{Z}$-span of a unique positive integer vector.

Definition 5.1 (Period vector). Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected critical network. The *period vector* $\mathbf{r}$ of $\mathcal{N}$ is the unique positive vector that generates $\mathcal{E} \cap K$. $\triangle$

The period vectors of some critical networks are shown in Table 5.1.

Remark. We would like to warn the reader about the difference between the period vector in this dissertation and in [12, 34]. For the sandpile network on a strongly connected digraph G, the period vector in [12, 34] is

$$\mathbf{r} = \left(\frac{t(G,v)}{\gcd_{w\in V}(t(G,w))}\right)_{v\in V},$$

where $t(G,v)$ is the number of directed spanning trees of G rooted toward v. On the other hand, the period vector based on our definition is

$$\mathbf{r} = \left(\frac{\mathrm{outdeg}(v)t(G,v)}{\gcd_{w\in V}(t(G,w))}\right)_{v\in V}. \tag{5.1}$$

This is because the former is the period vector for the Laplacian matrix Δ_G, while the latter is the period vector for the production matrix (which in this case is equal to $A_G D_G^{-1}$). $\triangle$

Recall the definition of $\rightarrow\leftarrow$ from Definition 4.6.

Definition 5.2 (Recurrent configuration). Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected critical network. A configuration $\mathbf{x}.\mathbf{q}$ is *recurrent* if both of the following conditions hold:

(i) There exists a nonempty legal execution for $\mathbf{x}.\mathbf{q}$; and

(ii) For all configurations $\mathbf{x}'.\mathbf{q}'$ satisfying $\mathbf{x}.\mathbf{q} \rightarrow\leftarrow \mathbf{x}'.\mathbf{q}'$, we have $\mathbf{x}'.\mathbf{q}' \longrightarrow \mathbf{x}.\mathbf{q}$.

$\triangle$

Later in Lemma 5.19 we relate recurrent configurations to recurrent components from §4.3.

Table 5.1: A list of the period vectors and exchange rate vectors of some critical networks. Note that $t(G, v)$ is the number of directed spanning trees rooted toward v, and $t^*(G, v)$ is the number of directed spanning trees rooted away from v.

Critical network $\mathcal{N}$ on G	Period vector $\mathbf{r}$ (Definition 5.1)	Exchange rate vector $\mathbf{s}$ (Definition 5.13)
Height-arrow network	$\left(\frac{\operatorname{outdeg}(v)t(G,v)}{\gcd_{w\in V}(t(G,w))}\right)_{v\in V}$	$\mathbf{1}$
Row chip-firing network	$(\operatorname{indeg}(v))_{v\in V}$	$\left(\frac{t^*(G,v)}{\gcd_{w\in V}(t^*(G,w))}\right)_{v\in V}$
Arithmetical network $(\mathcal{D}, \mathcal{M}, \mathbf{b})$	$\mathcal{D}\mathbf{b}$	depends on $\mathcal{M}$
McKay-Cartan network of $(\mathcal{G}, \gamma)$	$(\dim\gamma \dim\chi)_{\chi\in\operatorname{Irrep}(\mathcal{G})}$	$(\dim\chi)_{\chi\in\operatorname{Irrep}(\mathcal{G})}$
Inverse network	depends on $\mathcal{N}$	$\mathbf{1}$

In the next lemma we give two other equivalent definitions for recurrent configurations. Recall that, for any $w \in A^*$, we denote by $|w|$ the vector in $\mathbb{N}^A$ that counts the number of occurrences of each letter in w. Also recall the definition of $w \setminus \mathbf{n}$ ($\mathbf{n} \in \mathbb{N}^A$) from Definition 4.1.

Lemma 5.3. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network, and let $\mathbf{x}.\mathbf{q}$ be a configuration of $\mathcal{N}$. The following are equivalent:*

(i) $\mathbf{x}.\mathbf{q}$ *is recurrent.*

(ii) *There exists a nonempty word $v \in A^*$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{v} \mathbf{x}.\mathbf{q}$.*

(iii) *There exists a legal execution w for $\mathbf{x}.\mathbf{q}$ such that $|w| = \mathbf{r}$ and $t_w\mathbf{q} = \mathbf{q}$.*

Proof. (i) implies (ii): By the first condition of recurrence, there is a nonempty word w' and a configuration $\mathbf{x}'.\mathbf{q}'$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{w'} \mathbf{x}'.\mathbf{q}'$. Since $\mathbf{x}.\mathbf{q}$ is recurrent, there exists $w'' \in A^*$ such that $\mathbf{x}'.\mathbf{q}' \xrightarrow{w''} \mathbf{x}.\mathbf{q}$. Then $w'w''$ is a nonempty word such that $\mathbf{x}.\mathbf{q} \xrightarrow{w'w''} \mathbf{x}.\mathbf{q}$, as desired.

(ii) implies (iii): By Lemma 3.9, the word v in (ii) satisfies $|v| \in K$ and $(I - P)|v| = \mathbf{M}_v(\mathbf{q}) = \mathbf{x} - \mathbf{x} = \mathbf{0}$. By the definition of period vector, it follows that $|v| = k\mathbf{r}$ for some positive k. In particular $|v|$ is a positive vector, and hence $\mathbf{q}$ is locally recurrent by Lemma 3.4(ii).

Write $w := v \setminus (k-1)\mathbf{r}$. The removal lemma (Lemma 4.2) implies that $\pi_{(k-1)\mathbf{r}}(\mathbf{x}.\mathbf{q}) \xrightarrow{w} \mathbf{x}.\mathbf{q}$. Note that $\pi_{(k-1)\mathbf{r}}(\mathbf{x}.\mathbf{q}) = \mathbf{x}.\mathbf{q}$ (since $\mathbf{r} \in K$ and $\mathbf{q}$ is locally recurrent), $|w| = \mathbf{r}$, and $t_w\mathbf{q} = t_{\mathbf{r}}\mathbf{q} = \mathbf{q}$. This proves the claim.

(iii) implies (i): It suffices to show that if there exist $w_1, w_2 \in A^*$ and $\mathbf{x}'.\mathbf{q}'$, $\mathbf{x}''.\mathbf{q}''$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{w_1} \mathbf{x}''.\mathbf{q}''$ and $\mathbf{x}'.\mathbf{q}' \xrightarrow{w_2} \mathbf{x}''.\mathbf{q}''$, then $\mathbf{x}'.\mathbf{q}' \longrightarrow \mathbf{x}.\mathbf{q}$.

Let k be a positive integer such that $k|w| = k\mathbf{r} \geq |w_1|$. (Note that k exists because $\mathbf{r} \geq \mathbf{1}$.) By the removal lemma,

$$\begin{tikzcd}
\mathbf{x}.\mathbf{q} \arrow[rr, "w^k"] \arrow[dr, "w_1"'] & & \mathbf{x}.\mathbf{q} \\
\mathbf{x}'.\mathbf{q}' \arrow[r, "w_2"] & \mathbf{x}''.\mathbf{q}'' \arrow[ur, "w^k \setminus |w_1|"'] &
\end{tikzcd}.$$

This shows that $\mathbf{x}'.\mathbf{q}' \longrightarrow \mathbf{x}.\mathbf{q}$, as desired. $\square$

Remark. Note that condition (ii) in Lemma 5.3 is used in [43, Definition 3.2] and [47, Definition 13] as the definition of recurrent configurations for sinkless rotor networks and for sinkless sandpile networks on a strongly connected digraph, respectively. If we drop the assumption that $\mathcal{N}$ is strongly connected, then condition (i) in Lemma 5.3 is strictly stronger than condition (ii), and condition (iii) is not well-defined as the period vector $\mathbf{r}$ is not unique.

In the next lemma, we list several basic properties of recurrent configurations.

Lemma 5.4. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network, and let $\mathbf{x}.\mathbf{q}$ be a recurrent configuration of $\mathcal{N}$. Then:*

(i) *The state $\mathbf{q}$ is locally recurrent.*

(ii) *The vector $\mathbf{x}$ is in $\mathbb{N}^A \setminus \{\mathbf{0}\}$.*

(iii) *The configuration $(\mathbf{x}+\mathbf{n}).\mathbf{q}$ is recurrent for all $\mathbf{n} \in \mathbb{N}^A$.*

(iv) *If $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$, then $\mathbf{x}'.\mathbf{q}'$ is also a recurrent configuration.*

Proof. (i) By Lemma 5.3(iii), there is a positive vector $\mathbf{w}$ such that $t_{\mathbf{w}}\mathbf{q} = \mathbf{q}$. By Lemma 3.4(ii), the state $\mathbf{q}$ is locally recurrent.

(ii) By Lemma 5.3(iii), there exists $w \in A^*$ such that $|w| \geq \mathbf{1}$ and $\pi_w(\mathbf{x}.\mathbf{q}) = \mathbf{x}.\mathbf{q}$. By Lemma 3.3(iii), the vector $\mathbf{x}$ is nonnegative. Since w is a nonempty legal execution on $\mathbf{x}.\mathbf{q}$, the vector $\mathbf{x}$ is nonzero.

(iii) By Lemma 5.3(ii), there is a nonempty word $w \in A^*$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}.\mathbf{q}$. By Lemma 3.3(ii) $(\mathbf{x}+\mathbf{n}).\mathbf{q} \xrightarrow{w} (\mathbf{x}+\mathbf{n}).\mathbf{q}$, and hence $(\mathbf{x}+\mathbf{n}).\mathbf{q}$ is recurrent by Lemma 5.3(ii).

(iv) Let $w_1 \in A^*$ be a word such that $\mathbf{x}.\mathbf{q} \xrightarrow{w_1} \mathbf{x}'.\mathbf{q}'$. By the definition of recurrence there exists $w_2 \in A^*$ such that $\mathbf{x}'.\mathbf{q}' \xrightarrow{w_2} \mathbf{x}.\mathbf{q}$. By Lemma 5.3(ii) there is a nonempty word $w_3 \in A^*$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{w_3} \mathbf{x}.\mathbf{q}$. Now note that $w_2w_3w_1$ is a nonempty word and $\mathbf{x}'.\mathbf{q}' \xrightarrow{w_2w_3w_1} \mathbf{x}'.\mathbf{q}'$. Hence $\mathbf{x}'.\mathbf{q}'$ is recurrent by Lemma 5.3(ii). $\square$

Here we present a consequence of Lemma 5.3 and Lemma 5.4 that will be used in Chapter 8. For any $a \in A$ we say that a word w is *a-tight* if $|w| \leq \mathbf{r}$ and $|w|(a) = \mathbf{r}(a)$.

Lemma 5.5. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. A configuration* $\mathbf{x}.\mathbf{q}$ *is recurrent if and only if these two conditions are satisfied:*

(i) *The state* $\mathbf{q}$ *is locally recurrent; and*

(ii) *For each $a \in A$ there exists an a-tight legal execution for* $\mathbf{x}.\mathbf{q}$.

Proof. Proof of only if direction: Condition (i) follows from Lemma 5.4(i). For condition (ii), Lemma 5.3(iii) implies that there exists a legal execution w for $\mathbf{x}.\mathbf{q}$ such that $|w| = \mathbf{r}$. Note that this w is an a-tight word for all $a \in A$, and condition (ii) follows.

Proof of if direction: For each $a \in A$ let w_a be an a-tight legal execution for $\mathbf{x}.\mathbf{q}$ given by condition (ii). By applying the exchange lemma (Lemma 4.4) consecutively, there exists a legal execution w for $\mathbf{x}.\mathbf{q}$ such that $|w| = \max_{a \in A}\{|w_a|\}$. The tightness condition for all $a \in A$ then implies that $|w| = \mathbf{r}$. Since $\mathbf{q}$ is locally recurrent by condition (i), we then have $t_w \mathbf{q} = t_{\mathbf{r}} \mathbf{q} = \mathbf{q}$. By Lemma 5.3(iii), we conclude that $\mathbf{x}.\mathbf{q}$ is recurrent. $\square$

We now outline a recurrence test for configurations of critical networks, answering a question posed in [17]. This recurrence test is called the *burning test*, named after a similar test for sandpile networks [30, 64, 1].

Given a configuration $\mathbf{x}.\mathbf{q}$ and a legal execution w for $\mathbf{x}.\mathbf{q}$, we say that w is $\mathbf{r}$-*maximal* if

(i) $|w| \leq \mathbf{r}$; and

(ii) For all $a \in A$ either $|w|(a) = \mathbf{r}(a)$ or wa is not a legal execution for $\mathbf{x}.\mathbf{q}$.

Theorem 5.6 (Critical burning test). *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. Let $\mathbf{x}.\mathbf{q}$ be a configuration of $\mathcal{N}$, and let $w \in A^*$ be any $\mathbf{r}$-maximal legal execution for $\mathbf{x}.\mathbf{q}$. Then $\mathbf{x}.\mathbf{q}$ is recurrent if and only if the word w satisfies $|w| = \mathbf{r}$ and $t_w\mathbf{q} = \mathbf{q}$.*

Proof. Proof of if direction: This follows directly from Lemma 5.3(iii).

Proof of only if direction: We first show that $|w| = \mathbf{r}$. By Lemma 5.3(iii) there is a legal execution w' for $\mathbf{x}.\mathbf{q}$ such that $|w'| = \mathbf{r}$. By the removal lemma (Lemma 4.2), the word $w' \setminus |w|$ is a legal execution for $\pi_w(\mathbf{x}.\mathbf{q})$. By the $\mathbf{r}$-maximality of w, we then have $w' \setminus |w|$ is the empty word, and hence $|w| = |w'| = \mathbf{r}$.

By Lemma 5.4(i) the state $\mathbf{q}$ is locally recurrent; hence $t_w\mathbf{q} = t_\mathbf{r}\mathbf{q} = \mathbf{q}$. The proof is now complete. □

Using Theorem 5.6, we derive a recurrence test for critical networks by constructing an $\mathbf{r}$-maximal legal execution w for $\mathbf{x}.\mathbf{q}$. The test in its precise form is given in Algorithm 1. See Figure 5.1 for an example of the burning test in action.

The running time of this burning test is equal to the sum of the entries of the period vector $\mathbf{r}$, which can take exponential time with respect to $|A|$ (One example is the sandpile network on a bidirected path with edge multiplicities 2 to the left and 3 to the right; see [34, Figure 1]). A related problem called the *reachability problem* (i.e., given two configurations $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$, decide whether $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$) is known to be in co-NP for the sinkless sandpile network on directed graphs [47].

In §8.1, we present a more efficient recurrence test called the "cycle test" for a subclass of critical networks called agent networks.

Input : A critical network $\mathcal{N}$, a configuration $\mathbf{x}.\mathbf{q}$.

Output: TRUE if $\mathbf{x}.\mathbf{q}$ is recurrent, FALSE if $\mathbf{x}.\mathbf{q}$ is not recurrent.

$\mathbf{q}' := \mathbf{q}$;

$\mathbf{x}' := \mathbf{x}$;

$w := \varnothing$;

while $|w|(a) < \mathbf{r}(a)$ *and* $\mathbf{x}'(a) \geq 1$ *for some* $a \in A$ **do**

    $\mathbf{x}' := \mathbf{x}' + \mathbf{M}(a, \mathbf{q}') - |a|$;

    $\mathbf{q}' := t_a \mathbf{q}'$;

    $w := wa$.

end

if $|w| == \mathbf{r}$ *and* $\mathbf{q} == \mathbf{q}'$ **then**

    output TRUE.

else

    output FALSE.

end

Algorithm 1: The burning test to check for recurrence of a configuration in a critical abelian network.

5.2 Thief networks of a critical network

In this section we relate the burning test for critical networks (Algorithm 1) to the preexisting burning test for subcritical networks. A *burning vector* $\mathbf{k}$ of $\mathcal{N}$ is a vector in the total kernel K that satisfies $\mathbf{k} \geq \mathbf{1}$ and $P\mathbf{k} \leq \mathbf{k}$.

Theorem 5.7 (Subcritical burning test [17, Theorem 5.5]). *Let $\mathcal{S}$ be a finite, locally irreducible, and subcritical abelian network with total kernel K and production matrix P. Let $\mathbf{k}$ be any burning vector of $\mathcal{N}$. Then $\mathbf{q} \in Q$ is recurrent*

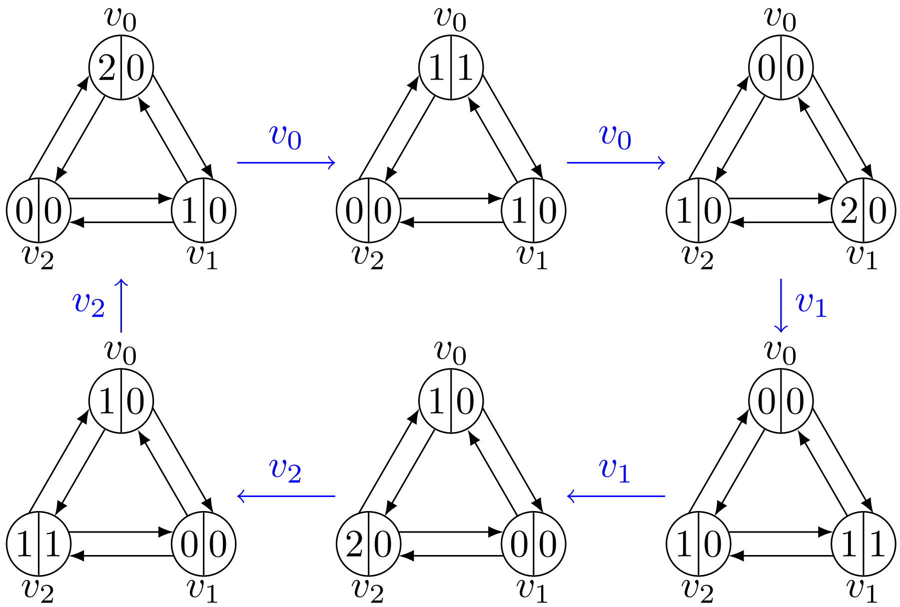

Figure 5.1: An instance of the burning test for the sinkless sandpile network on the bidirected cyle C_3. In the figure, the left part of $v \in V$ records $\mathbf{x}(v)$, while the right part records $\mathbf{q}(v)$. The inputs are $\mathbf{x} := (2,1,0)^\top$, $\mathbf{q} := (0,0,0)^\top$, and $\mathbf{r} = (2,2,2)^\top$. The configuration $\mathbf{x}.\mathbf{q}$ is recurrent by the burning test.

if and only if $(I-P)\mathbf{k}.\mathbf{q} \longrightarrow \mathbf{0}.\mathbf{q}$. $\square$

See Figure 5.2 for an example of this burning test for sandpile networks with sinks.

The relation between these two burning tests can be explained by using the notion of thief networks.

Remark. In this section we often discuss two abelian networks at the same time. When there is more than one network in the discussion, we will indicate in the notation which network we are referring to, e.g. $t_a^{\mathcal{N}}$, $\mathbf{M}_a^{\mathcal{N}}$, $\pi_a^{\mathcal{N}}$, $\mathcal{N}$-recurrent, $\underset{\mathcal{N}}{\longrightarrow}$, etc.

For $R \subseteq A$ and $\mathbf{x} \in \mathbb{Z}^A$, let $\mathbf{x}_R$ denote the vector in $\mathbb{Z}^A$ for which $\mathbf{x}_R(a) := \mathbf{x}(a)$

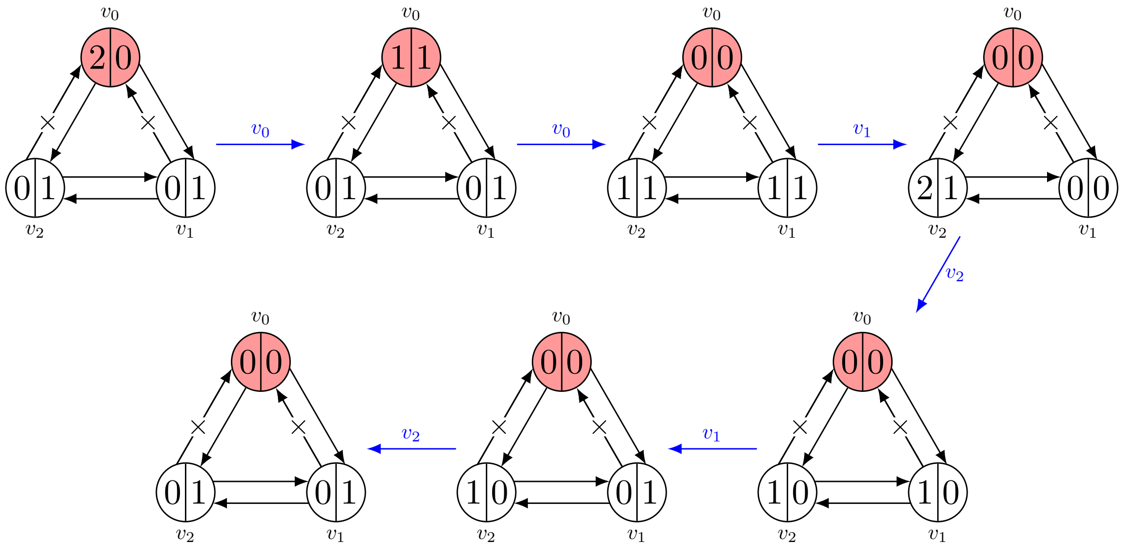

Figure 5.2: A subcritical burning test for the sandpile network with sink at $S = \{v_0\}$ (colored in red). In the figure, the left part of $v \in V$ records $\mathbf{x}(v)$, while the right part records $\mathbf{q}(v)$. The inputs for the test are $\mathbf{q} := (0, 1, 1)^\top$ and $\mathbf{k} := (2, 2, 2)^\top$. (Note that $(I - P)\mathbf{k} = (2, 0, 0)^\top$ here.) The state $\mathbf{q}$ is recurrent by the burning test.

if $a \in R$ and $\mathbf{x}_R(a) := 0$ if $a \notin R$.

Let $\mathcal{N}$ be an abelian network, and let $R \subseteq A$. The *thief network based on* $\mathcal{N}$ *with messages restricted to* R (*thief network* $\mathcal{N}_R$ for short) is the abelian network (with the same underlying digraph as $\mathcal{N}$) defined by:

- The alphabet $A^{\mathcal{N}_R}$, the state space $Q^{\mathcal{N}_R}$ and the transition functions $(t_a^{\mathcal{N}_R})_{a \in A}$ of $\mathcal{N}_R$ are identical with those of $\mathcal{N}$.
- For any $a \in A$ and $\mathbf{q} \in Q$, the message-passing vector $\mathbf{M}_a^{\mathcal{N}_R}(\mathbf{q})$ is equal to $(\mathbf{M}_a^{\mathcal{N}}(\mathbf{q}))_R$.

One can think of $\mathcal{N}_R$ as a network of computers where the wires used for transmitting letters from $A \setminus R$ are stolen by a wire thief. Hence all the letters from $A \setminus R$ will not appear in the messages exchanged between computers in the network.

Note that $t_a^{\mathcal{N}_R}$ and $\mathbf{M}_a^{\mathcal{N}_R}$ are defined even for $a \in A \setminus R$, so $\mathcal{N}_R$ retains the ability to process inputs with letters from $A \setminus R$. One can think of this to mean that the keyboards for the computers in the network are working fine and are not tampered by the wire thief.

The reader can use height-arrow networks with sinks (Example 3.14) as a running example when reading this section. Note that a height-arrow network with sinks at S (Example 3.14) is the thief network of the corresponding sinkless height-arrow network (Example 3.13) restricted to $V \setminus S$.

We now relate the total kernel and the production matrix of $\mathcal{N}_R$ to those of $\mathcal{N}$.

Let M be a matrix with rows indexed by A. For $R \subseteq A$, we denote by M_R the matrix obtained by replacing the rows of M indexed by $A \setminus R$ with the zero vector.

Lemma 5.8. *Let $\mathcal{N}$ be a finite and locally irreducible abelian network with total kernel K and production matrix P, and let $R \subseteq A$.*

(i) *The network $\mathcal{N}_R$ is finite and locally irreducible, the total kernel of $\mathcal{N}_R$ is equal to K, and the production matrix of $\mathcal{N}_R$ is equal to P_R.*

(ii) *If $\mathcal{N}$ is a strongly connected critical network and $R \subsetneq A$, then $\mathcal{N}_R$ is a subcritical network.*

Proof. (i) Since the transition functions of $\mathcal{N}_R$ are the same as those of $\mathcal{N}$, the network $\mathcal{N}_R$ is finite and locally irreducible. By the same reason, the total kernel of $\mathcal{N}_R$ is equal to K.

Since $\mathbf{M}_a^{\mathcal{N}_R}(\mathbf{q}) = (\mathbf{M}_a^{\mathcal{N}}(\mathbf{q}))_R$ for all $a \in A$ and $\mathbf{q} \in Q$, it follows directly from the definition that the production matrix of $\mathcal{N}_R$ is equal to P_R.

(ii) Note that P is strongly connected (since $\mathcal{N}$ is strongly connected), $P_R \leq P$ (by definition), and $P_R \neq P$ (since $R \subsetneq A$). The claim now follows directly from the Perron-Frobenius theorem (Lemma 3.10(iv)). $\square$

We remark that the network $\mathcal{N}_R$ is not strongly connected whenever $R \subsetneq A$, as some of the rows of P_R are zero vectors.

Recall the definition of recurrent configurations for a critical network (Definition 5.2) and the definition of recurrent states for a subcritical network (Definition 4.26). We now state the main results of this subsection, which are two propositions that relate the recurrent configurations of a critical network to the recurrent states of its thief networks.

Recall that the support of $\mathbf{x} \in \mathbb{Z}^A$ is $\text{supp}(\mathbf{x}) = \{a \in A : \mathbf{x}(a) \neq 0\}$.

Proposition 5.9. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. Let $\mathbf{x} \in \mathbb{N}^A \setminus \{\mathbf{0}\}$ and let $R := A \setminus \text{supp}(\mathbf{x})$. If $\mathbf{x}.\mathbf{q}$ is an $\mathcal{N}$-recurrent configuration, then $\mathbf{q}$ is an $\mathcal{N}_R$-recurrent state.*

We remark that the converse of Proposition 5.9 is false; see Example 5.10. With that being said, we will present a special family of critical networks for which the converse holds in Lemma 8.12.

Example 5.10. Let $\mathcal{N}$ be the sinkless sandpile network (Example 3.12) on the bidirected cycle C_3, and let $R := V \setminus \{v_0\}$.

Let $\mathbf{x} \in \mathbb{Z}^V$ and $\mathbf{q} \in (\mathbb{Z}_2)^V$ be given by:

$$\mathbf{x} := (1,0,0)^\top \quad \text{and} \quad \mathbf{q} := (0,1,1)^\top.$$

The state $\mathbf{q}$ is $\mathcal{N}_R$-recurrent because it passes the burning test in Theorem 5.7, as shown in Figure 5.2. On the other hand, note that $\mathbf{x}.\mathbf{q} \xrightarrow[\mathcal{N}]{v_0} \mathbf{0}.\mathbf{q}'$, where $\mathbf{q}' := (1,1,1)^\top$. This shows that $\mathbf{x}.\mathbf{q}$ is an $\mathcal{N}$-halting configuration, and hence $\mathbf{x}.\mathbf{q}$ is not $\mathcal{N}$-recurrent. $\triangle$

Recall that $\mathbf{r}$ denotes the period vector of a critical network $\mathcal{N}$ (Definition 5.1).

Proposition 5.11. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network, and let $R \subsetneq A$. Then $\mathbf{q} \in Q$ is an $\mathcal{N}_R$-recurrent state if and only if $(I - P_R)\mathbf{r}.\mathbf{q}$ is an $\mathcal{N}$-recurrent configuration.*

In particular, checking for the recurrence of $\mathbf{q} \in Q$ in $\mathcal{N}_R$ can be done by applying the critical burning test for $\mathcal{N}$ (Algorithm 1) on $(I - P_R)\mathbf{r}.\mathbf{q}$, and it can be shown that this test is equivalent to the subcritical burning test for $\mathcal{N}_R$ (Theorem 5.7) with $\mathbf{k} = \mathbf{r}$. The critical burning test for $\mathcal{N}$ on $(I - P)\mathbf{r}.\mathbf{q}$ can be derived from the subcritical burning test for $\mathcal{N}_R$ in a similar manner.

We now build towards the proof of these two propositions, and we start with a technical lemma.

Lemma 5.12. *Let $\mathcal{N}$ be an abelian network and let $R \subseteq A$.*

(i) *If $w \in A^*$ is an $\mathcal{N}_R$-legal execution for* $\mathbf{x}.\mathbf{q}$*, then w is an $\mathcal{N}$-legal execution for* $\mathbf{x}.\mathbf{q}$.

(ii) *If $w \in A^*$ is an $\mathcal{N}$-legal execution for* $\mathbf{x}.\mathbf{q}$*, then w is an $\mathcal{N}_R$-legal execution for* $(\mathbf{x}_R + \mathbf{w}_{A \setminus R}).\mathbf{q}$*, where* $\mathbf{w} := |w|$.

Proof. Part (i) follows from the inequality $\mathbf{M}_a^{\mathcal{N}_R}(\mathbf{q}) \leq \mathbf{M}_a^{\mathcal{N}}(\mathbf{q})$ for all $a \in A$ and $\mathbf{q} \in Q$.

We now prove part (ii). Let $w = a_1 \cdots a_\ell$. For any $i \in \{0, 1, \ldots, \ell\}$ we write

$$w_i := a_1 \ldots a_i, \quad \mathbf{x}_i.\mathbf{q}_i := \pi^{\mathcal{N}}_{a_1 \cdots a_i}(\mathbf{x}.\mathbf{q}), \quad \mathbf{x}'_i.\mathbf{q}'_i := \pi^{\mathcal{N}_R}_{a_1 \cdots a_i}((\mathbf{x}_R + \mathbf{w}_{A \setminus R}).\mathbf{q}).$$

It suffices to show that $\mathbf{x}'_{i-1}(a_i) \geq 1$ for all $i \in \{1, \ldots, \ell\}$.

Fix $i \in \{1, \ldots, \ell\}$. Then

$$\begin{aligned}
\mathbf{x}'_{i-1}(a_i) &= \mathbf{x}_R(a_i) + \mathbf{w}_{A \setminus R}(a_i) + \mathbf{M}^{\mathcal{N}_R}_{w_{i-1}}(\mathbf{q})(a_i) - |w_{i-1}|(a_i) \\
&= \begin{cases} |w|(a_i) - |w_{i-1}|(a_i) & \text{if } a_i \in A \setminus R; \\ \mathbf{x}(a_i) + \mathbf{M}^{\mathcal{N}}_{w_{i-1}}(\mathbf{q})(a_i) - |w_{i-1}|(a_i) = \mathbf{x}_{i-1}(a_i) & \text{if } a_i \in R. \end{cases}
\end{aligned}$$

Note that $|w|(a_i) - |w_{i-1}|(a_i) \geq 1$ because the i-th letter of w is a_i. Also note that $\mathbf{x}_{i-1}(a_i) \geq 1$ because w is legal for $\mathbf{x}.\mathbf{q}$. Hence we conclude that $\mathbf{x}'_{i-1}(a_i) \geq 1$, as desired. □

Proof of Proposition 5.9. Note that by Lemma 5.8(ii) the network $\mathcal{N}_R$ is subcritical since $R \subsetneq A$. Also note that the period vector $\mathbf{r}$ of $\mathcal{N}$ satisfies $\mathbf{r} \in K$, $\mathbf{r} \geq \mathbf{1}$, and $P_R\mathbf{r} = \mathbf{r}_R \leq \mathbf{r}$. Hence by Theorem 5.7 it suffices to show that $(I - P_R)\mathbf{r}.\mathbf{q} \xrightarrow[\mathcal{N}_R]{} \mathbf{0}.\mathbf{q}$.

Since $\mathbf{x}.\mathbf{q}$ is $\mathcal{N}$-recurrent, by Theorem 5.6 there exists $w \in A^*$ such that $\mathbf{x}.\mathbf{q} \xrightarrow[\mathcal{N}]{w} \mathbf{x}.\mathbf{q}$ and $|w| = \mathbf{r}$. Since $\mathbf{x}_R = \mathbf{0}$ by assumption, the word w is an $\mathcal{N}_R$-legal execution for $\mathbf{r}_{A \setminus R}.\mathbf{q}$ by Lemma 5.12(ii). Now note that

$$\begin{aligned}
\pi^{\mathcal{N}_R}_w(\mathbf{r}_{A \setminus R}.\mathbf{q}) &= (\mathbf{r}_{A \setminus R} + \mathbf{M}^{\mathcal{N}_R}_w(\mathbf{q}) - |w|).t_w\mathbf{q} \\
&= (\mathbf{r}_{A \setminus R} + (\mathbf{M}^{\mathcal{N}}_w(\mathbf{q}))_R - \mathbf{r}).\mathbf{q} \\
&= \mathbf{0}.\mathbf{q} \quad (\text{because } \mathbf{M}^{\mathcal{N}}_w(\mathbf{q}) = \mathbf{r}).
\end{aligned}$$

Also note that $\mathbf{r}_{A \setminus R} = (I - P_R)\mathbf{r}$. Hence, we conclude that $(I - P_R)\mathbf{r}.\mathbf{q} \xrightarrow[\mathcal{N}_R]{w} \mathbf{0}.\mathbf{q}$, as desired. □

Proof of Proposition 5.11. The if direction follows from Proposition 5.9 and the fact that $\operatorname{supp}((I - P_R)\mathbf{r}) = A \setminus R$.

We now prove the only if direction. Since $\mathbf{q}$ is $\mathcal{N}_R$-recurrrent, by Theorem 5.7 there exists $w \in A^*$ such that $(I - P_R)\mathbf{r}.\mathbf{q} \xrightarrow[\mathcal{N}_R]{w} \mathbf{0}.\mathbf{q}$. By Lemma 3.9, this implies that $\mathbf{M}_w^{\mathcal{N}_R}(\mathbf{q}) = P_R|w|$. Then

$$(I - P_R)\mathbf{r} = |w| - \mathbf{M}_w^{\mathcal{N}_R}(\mathbf{q}) = (I - P_R)|w|. \tag{5.2}$$

Since P_R has spectral radius strictly less than 1 (by Lemma 5.8(ii)), the matrix $I - P_R$ is invertible. It then follows from (5.2) that $|w| = \mathbf{r}$.

By Lemma 5.12(i), the word w is an $\mathcal{N}$-legal execution for $(I - P_R)\mathbf{r}.\mathbf{q}$. Since $|w| = \mathbf{r}$ and $t_{\mathbf{r}}^{\mathcal{N}}\mathbf{q} = t_w^{\mathcal{N}_R}\mathbf{q} = \mathbf{q}$, by Theorem 5.6 we conclude that $(I - P_R)\mathbf{r}.\mathbf{q}$ is an $\mathcal{N}$-recurrent configuration, as desired. □

5.3 The capacity and the level of a configuration

In this section we define the capacity of a network and the level of a configuration of a critical network. Those two notions will be used later in §5.4 to give a combinatorial description for the invertible recurrent components of a critical network.

Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected abelian network. By the Perron-Frobenius theorem (Lemma 3.10(v)) the $\lambda(P^\top)$-eigenspace of $P^\top$ is spanned by a positive real vector.

Definition 5.13 (Exchange rate vector). Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected abelian network. An *exchange rate vector* $\mathbf{s}$ is a positive real vector that spans the $\lambda(P^\top)$-eigenspace of $P^\top$. △

The vector $\mathbf{s}$ measures the comparative value between any two letters in $\mathcal{N}$, in a manner to be made precise soon.

Throughout this dissertation we fix an exchange rate vector $\mathbf{s}$. In the case when $\lambda(P) = \lambda(P^\top)$ is rational, then we choose $\mathbf{s}$ to be an exchange rate vector that is a positive integer vector and such that $\gcd_{a \in A} \mathbf{s}(a) = 1$. This choice of $\mathbf{s}$ exists and is unique by the Perron-Frobenius theorem (Lemma 3.10(vi)). The exchange rate vectors of some critical networks are shown in Table 5.1.

Recall that a configuration $\mathbf{x}.\mathbf{q}$ *halts* if $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$ for some $\mathbf{x}' \leq \mathbf{0}$ and some $\mathbf{q}' \in Q$.

Definition 5.14 (Capacity). Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected abelian network. The *capacity* of a configuration $\mathbf{x}.\mathbf{q}$ and the *capacity* of a state $\mathbf{q}$ are

$$\operatorname{cap}(\mathbf{x}.\mathbf{q}) := \sup_{\mathbf{z} \in \mathbb{Z}^A} \{\mathbf{s}^\top \mathbf{z} : (\mathbf{z}+\mathbf{x}).\mathbf{q} \text{ halts}\}; \qquad \operatorname{cap}(\mathbf{q}) := \operatorname{cap}(\mathbf{0}.\mathbf{q}),$$

respectively. The *capacity* of $\mathcal{N}$ is

$$\operatorname{cap}(\mathcal{N}) := \max_{\mathbf{q} \in Q} \{\operatorname{cap}(\mathbf{q})\}. \qquad \triangle$$

In words, the capacity of a configuration is the maximum number of letters (weighted according to the exchange rate vector) that can be absorbed by the configuration without causing the process to run forever.

The following is an example that illustrates the notion of capacity.

Example 5.15. First consider the sinkless rotor network (Example 3.11). In this network, processing a chip will result in moving the chip to another vertex of the digraph. So if there are a positive number of chips in the network, then the process will run forever, as at any time stage there will always be some chips that can be moved around. Hence the capacity of a sinkless rotor network is equal to zero.

On the other end of the scale, we have sinkless sandpile networks (Example 3.12). In this network, processing a chip means either moving the chip into the locker $\mathcal{P}_v$ (if $\mathcal{P}_v$ is not full), or sending all stored chips in $\mathcal{P}_v$ together with the processed chip to other vertices (if $\mathcal{P}_v$ is already full). Note that each locker $\mathcal{P}_v$ can store at most $\mathrm{outdeg}(v) - 1$ chips. Therefore, if the total number of chips is strictly greater than $|E| - |V| = \sum_{v \in V}(\mathrm{outdeg}(v) - 1)$, then at any time stage of the process there is always an unstored chip that can be processed. Hence the sandpile network has capacity at most $|E|-|V|$. On the other hand, the configuration $\mathbf{x}.\mathbf{q}$ with $\mathbf{x} := (\mathrm{outdeg}(v)-1)_{v\in V}$ and $\mathbf{q} := \mathbf{0}$ is a halting configuration, which implies that the sandpile network has capacity at least $\mathbf{1}^\top \mathbf{x} = |E| - |V|$. Hence we conclude that the capacity of a sinkless sandpile network is equal to $|E| - |V|$.

By an analogous argument, the capacity of a height-arrow network is equal to $\sum_{v\in V}(\tau_v - 1)$, which lies between the capacity of rotor network and sandpile network on the same digraph. $\triangle$

The capacity of a subcritical network is infinite, as every configuration halts in a subcritical network (Theorem 4.24). We now show that conversely, the capacity of a critical or supercritical network is always finite.

Recall that a configuration $\mathbf{x}.\mathbf{q}$ is stable if $\mathbf{x} \leq \mathbf{0}$.

Lemma 5.16. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected abelian network. If $\mathcal{N}$ is a critical or supercritical network, then* $\mathrm{cap}(\mathcal{N}) < \infty$.

Proof. Suppose to the contrary that the claim is false. Then there exist configurations $\mathbf{z}_1.\mathbf{q}_1, \mathbf{z}_2.\mathbf{q}_2, \ldots$ and stable configurations $\mathbf{z}'_1.\mathbf{q}'_1, \mathbf{z}'_2.\mathbf{q}'_2, \ldots$ such that $\mathbf{z}_i.\mathbf{q}_i \xrightarrow{w_i} \mathbf{z}'_i.\mathbf{q}'_i$ for all $i \geq 1$ and $\mathbf{s}^\top \mathbf{z}_i \to \infty$ as $i \to \infty$.

By the pigeonhole principle, there exists an infinite subset J of $\mathbb{Z}_{\geq 1}$ such that $\mathbf{q}_j = \mathbf{q}_i$ and $\mathbf{q}'_j = \mathbf{q}'_i$ for all $i, j \in J$. Fix an $j \in J$ and write $\lambda := \lambda(P)$. Then for any $i \in J$,

$$\begin{aligned}
\mathbf{z}_i - \mathbf{z}_j &= (\mathbf{z}'_i - \mathbf{M}_{w_i}(\mathbf{q}_i) + |w_i|) - (\mathbf{z}'_j - \mathbf{M}_{w_j}(\mathbf{q}_i) + |w_j|) \\
&= (\mathbf{z}'_i + (I-P)|w_i|) - (\mathbf{z}'_j + (I-P)|w_j|) \quad \text{(by Lemma 3.9)}
\end{aligned}$$

Multiplying $\mathbf{s}^\top$ to both sides of the equation above, we get

$$\mathbf{s}^\top(\mathbf{z}_i - \mathbf{z}_j) = (\mathbf{s}^\top \mathbf{z}'_i + (1-\lambda)\mathbf{s}^\top |w_i|)) - (\mathbf{s}^\top \mathbf{z}'_j + (1-\lambda)\mathbf{s}^\top |w_j|) \tag{5.3}$$

Now note that $\mathbf{s}^\top \mathbf{z}'_i \leq 0$ since $\mathbf{z}'_i \leq \mathbf{0}$, and $(1-\lambda) \leq 0$ by assumption. Plugging this into (5.3), we get

$$\mathbf{s}^\top \mathbf{z}_i \leq \mathbf{s}^\top \mathbf{z}_j - \mathbf{s}^\top(\mathbf{z}'_j + (1-\lambda)|w_j|).$$

This gives an upper bound for $\mathbf{s}^\top \mathbf{z}_i$ that is independent of i, which contradicts the assumption that $s^\top \mathbf{z}_i \to \infty$ as $i \to \infty$. □

Definition 5.17 (Level). Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected critical network. The *level* of a state $\mathbf{q}$ and the *level* of a configuration $\mathbf{x}.\mathbf{q}$ are

$$\operatorname{lvl}(\mathbf{q}) := \operatorname{cap}(\mathcal{N}) - \operatorname{cap}(\mathbf{q}); \qquad \operatorname{lvl}(\mathbf{x}.\mathbf{q}) := \operatorname{cap}(\mathcal{N}) - \operatorname{cap}(\mathbf{x}.\mathbf{q}),$$

respectively. △

Note that by the definition of capacity, we have

$$\operatorname{lvl}(\mathbf{x}.\mathbf{q}) = \operatorname{cap}(\mathcal{N}) - \operatorname{cap}(\mathbf{q}) + \mathbf{s}^\top \mathbf{x} = \operatorname{lvl}(\mathbf{q}) + \mathbf{s}^\top \mathbf{x}.$$

For height-arrow networks, the level of a configuration $\mathbf{x}.\mathbf{q}$ is equal to $\sum_{v \in V} \mathbf{x}(v) + \mathbf{q}(v)$, the total number of chips (counting both stored and unstored chips) in the configuration.

Here we list basic properties of the capacity (equivalently, level) of a configuration in a critical network.

Lemma 5.18. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network.*

(i) *If $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ are configurations such that $\mathbf{x}.\mathbf{q} \dashrightarrow \mathbf{x}'.\mathbf{q}'$, then $\operatorname{cap}(\mathbf{x}'.\mathbf{q}') \leq \operatorname{cap}(\mathbf{x}.\mathbf{q})$.*

(ii) *If $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ are configurations such that $\mathbf{x}.\mathbf{q} \dashrightarrow \mathbf{x}'.\mathbf{q}'$ and $\mathbf{q} \in \operatorname{Loc}(\mathcal{N})$, then $\operatorname{cap}(\mathbf{x}.\mathbf{q}) = \operatorname{cap}(\mathbf{x}'.\mathbf{q}')$.*

(iii) *For any $\mathbf{q} \in Q$, we have $0 \leq \operatorname{cap}(\mathbf{q}) \leq \operatorname{cap}(\mathcal{N})$.*

(iv) *There exists $\mathbf{q} \in Q$ such that $\operatorname{cap}(\mathbf{q}) = \operatorname{cap}(\mathcal{N})$.*

(v) *There exists $\mathbf{q} \in \operatorname{Loc}(\mathcal{N})$ such that $\operatorname{cap}(\mathbf{q}) = 0$.*

Proof. (i) Let $\mathbf{z} \in \mathbb{Z}^A$ be any vector such that $(\mathbf{z}+\mathbf{x}').\mathbf{q}'$ halts. Then there exists a stable configuration $\mathbf{y}.\mathbf{p}$ such that $(\mathbf{z}+\mathbf{x}').\mathbf{q}' \dashrightarrow \mathbf{y}.\mathbf{p}$. By the transitivity of $\dashrightarrow$, we then have $(\mathbf{z}+\mathbf{x}).\mathbf{q} \dashrightarrow \mathbf{y}.\mathbf{p}$. By the least action principle (Corollary 4.3), we conclude that $(\mathbf{z}+\mathbf{x}).\mathbf{q}$ halts. Hence

$$\{\mathbf{z} \in \mathbb{Z}^A : (\mathbf{z}+\mathbf{x}').\mathbf{q}' \text{ halts}\} \subseteq \{\mathbf{z} \in \mathbb{Z}^A : (\mathbf{z}+\mathbf{x}).\mathbf{q} \text{ halts}\},$$

which implies that $\operatorname{cap}(\mathbf{x}'.\mathbf{q}') \leq \operatorname{cap}(\mathbf{x}.\mathbf{q})$.

(ii) By part (i), it suffices to show that $\operatorname{cap}(\mathbf{x}.\mathbf{q}) \leq \operatorname{cap}(\mathbf{x}'.\mathbf{q}')$. Let $w \in A^*$ be such that $\mathbf{x}.\mathbf{q} \overset{w}{\dashrightarrow} \mathbf{x}'.\mathbf{q}'$, and let k be such that $k\mathbf{r} \geq |w|$. (Note that k exists because the period vector $\mathbf{r}$ is positive.) Then

$$\pi_{k\mathbf{r}-|w|}(\mathbf{x}'.\mathbf{q}') = \pi_{k\mathbf{r}-|w|}(\pi_w(\mathbf{x}.\mathbf{q})) = \pi_{k\mathbf{r}}(\mathbf{x}.\mathbf{q}) = \mathbf{x}.\mathbf{q},$$

where the last equality is because $\mathbf{q}$ is locally recurrent. Hence we have $\mathbf{x}'.\mathbf{q}' \dashrightarrow \mathbf{x}.\mathbf{q}$, which then implies that $\mathrm{cap}(\mathbf{x}.\mathbf{q}) \leq \mathrm{cap}(\mathbf{x}'.\mathbf{q}')$ by part (i), as desired.

(iii) For any $\mathbf{q} \in Q$ the configuration $\mathbf{0}.\mathbf{q}$ halts by definition, and hence $\mathrm{cap}(\mathbf{q}) \geq 0$. The other inequality follows directly from the definition of $\mathrm{cap}(\mathcal{N})$.

(iv) This follows directly from the definition of $\mathrm{cap}(\mathcal{N})$.

(v) Let $\mathbf{q}$ be a locally recurrent state with minimum capacity among all locally recurrent states. Let $\mathbf{z} \in \mathbb{Z}^A$ be any vector such that $\mathbf{z}.\mathbf{q}$ halts. By definition, there exists a stable configuration $\mathbf{y}.\mathbf{p}$ such that $\mathbf{z}.\mathbf{q} \longrightarrow \mathbf{y}.\mathbf{p}$ and $\mathbf{y} \leq \mathbf{0}$.

By Lemma 3.5(i), the state $\mathbf{p}$ is locally recurrent, and hence $\mathrm{cap}(\mathbf{q}) \leq \mathrm{cap}(\mathbf{p})$ by the minimality assumption. On the other hand, by part (ii) we have $-\mathbf{s}^\top\mathbf{z} + \mathrm{cap}(\mathbf{q}) = -\mathbf{s}^\top\mathbf{y} + \mathrm{cap}(\mathbf{p})$. These two facts then imply $\mathbf{s}^\top\mathbf{z} \leq \mathbf{0}$.

Since the choice of $\mathbf{z}$ is arbitrary, we conclude that $\mathrm{cap}(\mathbf{q}) \leq 0$. By part (iii) it then follows that $\mathrm{cap}(\mathbf{q}) = 0$. $\square$

Lemma 5.18(ii) implies that, in a critical network, the level of a configuration does not change over time, provided that the initial state of the configuration is locally recurrent. This distinguishes critical networks from subcritical and supercritical networks, where an analogous notion of level can decrease for the former, and increase for the latter.

5.4 Stoppable levels: When does the torsion group act transitively?

In this section we study the torsion group of a critical network in more detail.

We start with the relationship between recurrent components (Definition 4.8) and recurrent configurations (Definition 5.2) of a critical network.

Lemma 5.19. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. A component $\mathcal{C}$ of the trajectory digraph is a recurrent component if and only if $\mathcal{C}$ contains a recurrent configuration.*

Proof. Proof of if direction: Let $\mathbf{x}.\mathbf{q}$ be a recurrent configuration in $\mathcal{C}$. Let $\mathbf{r}$ be the period vector of $\mathcal{N}$ (Definition 5.1). By Lemma 5.3(iii), there exists $w \in A^*$ such that $|w| = \mathbf{r}$ and

$$\cdots \xrightarrow{w} \mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}.\mathbf{q} \xrightarrow{w} \cdots .$$

This is a diverse infinite walk (Definition 4.8) in $\mathcal{C}$ (because $|w| = \mathbf{r} \geq \mathbf{1}$), and hence $\mathcal{C}$ is a recurrent component.

Proof of only if direction: By Proposition 4.10, the recurrent component $\mathcal{C}$ contains a diverse cycle. In particular, this implies that there exists a configuration $\mathbf{x}.\mathbf{q}$ in $\mathcal{C}$ and a nonempty word w such that $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}.\mathbf{q}$. Now note that $\mathbf{x}.\mathbf{q}$ is a recurrent configuration by Lemma 5.3(ii). This proves the claim. □

Note that a recurrent component may contain a non-recurrent configuration, as shown in the following example.

Example 5.20. Consider the sinkless sandpile network $\mathcal{N}$ (Example 3.12) on the bidirected cycle C_3. Let $\mathbf{x} \in \mathbb{Z}^V$ and $\mathbf{q} \in (\mathbb{Z}_2)^V$ be given by:

$$\mathbf{x} := (2, 1, 0)^\top \quad \text{and} \quad \mathbf{q} := (0, 0, 0)^\top.$$

Note that $\mathbf{x}.\mathbf{q}$ is a recurrent configuration as it passes the burning test, as shown in Figure 5.1.

Let $\mathbf{x}' \in \mathbb{Z}^V$ and $\mathbf{q}' \in (\mathbb{Z}_2)^V$ be given by:

$$\mathbf{x}' := (1, 2, -1)^\top \quad \text{and} \quad \mathbf{q}' := (0, 1, 0)^\top.$$

The configuration $\mathbf{x}'.\mathbf{q}'$ is in the same component as $\mathbf{x}.\mathbf{q}$ since $\mathbf{x}'.\mathbf{q}' \xrightarrow{v_1} \mathbf{x}.\mathbf{q}$. However, $\mathbf{x}'.\mathbf{q}'$ is not recurrent by Lemma 5.4(ii) since $\mathbf{x}'$ has a negative entry. $\triangle$

The *level* of a recurrent component $\mathcal{C}$ is

$$\mathrm{lvl}(\mathcal{C}) := \mathrm{lvl}(\mathbf{x}.\mathbf{q}),$$

where $\mathbf{x}.\mathbf{q}$ is any recurrent configuration in $\mathcal{C}$. The value of $\mathrm{lvl}(\mathcal{C})$ does not depend on the choice of $\mathbf{x}.\mathbf{q}$ as a consequence of Lemma 5.18(ii) and Lemma 5.4(i). For any $m \in \mathbb{N}$ we denote by $\overline{\mathrm{Rec}}(\mathcal{N}, m)$ the set of recurrent components of $\mathcal{N}$ with level m.

Definition 5.21 (Stoppable level). Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected critical network. The set of *stoppable levels* of $\mathcal{N}$ is

$$\mathrm{Stop}(\mathcal{N}) := \{m \in \mathbb{N} \mid m = \mathrm{lvl}(\mathbf{x}.\mathbf{q}) \text{ for some } \mathbf{x} \leq \mathbf{0} \text{ and } \mathbf{q} \in \mathrm{Loc}(\mathcal{N})\}. \qquad \triangle$$

Example 5.22. Let $\mathcal{N}$ be the row chip-firing network (Example 3.15) from Figure 3.6. The underlying digraph G has two vertices v_1 and v_2, with three edges directed from v_1 to v_2, and two edges directed from v_2 to v_1.

The production matrix and the exchange rate vector of this network are given by

$$P = \begin{bmatrix} 0 & \frac{2}{3} \\ \frac{3}{2} & 0 \end{bmatrix}, \qquad \mathbf{s} = \begin{bmatrix} 3 \\ 2 \end{bmatrix},$$

respectively. The state space is $Q = \mathbb{Z}_2 \times \mathbb{Z}_3$, and the levels of the states are given by:

$$\begin{aligned} \mathrm{lvl}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}\right) &= 0, \quad \mathrm{lvl}\left(\begin{bmatrix} 0 \\ 1 \end{bmatrix}\right) = 2, \quad \mathrm{lvl}\left(\begin{bmatrix} 0 \\ 2 \end{bmatrix}\right) = 4, \\ \mathrm{lvl}\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}\right) &= 3, \quad \mathrm{lvl}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) = 5, \quad \mathrm{lvl}\left(\begin{bmatrix} 1 \\ 2 \end{bmatrix}\right) = 7. \end{aligned}$$

The capacity of this network is then equal to 7, and the set of stoppable levels is given by:

$$\mathrm{Stop}(\mathcal{N}) = \{0, 1, 2, 3, 4, 5, 7\}.$$

(Note that 1 is a stoppable level because the configuration $\begin{bmatrix} 0 \\ -1 \end{bmatrix} . \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ has level 1.) $\triangle$

Lemma 5.23. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. Then*

$$\mathrm{Stop}(\mathcal{N}) \subseteq \{0, 1, \dots, \mathrm{cap}(\mathcal{N})\},$$

with equality if the exchange rate vector $\mathbf{s}$ *has a coordinate equal to* 1.

Proof. Let $\mathbf{x}.\mathbf{q}$ be any configuration such that $\mathbf{x} \le \mathbf{0}$ and $\mathbf{q} \in \mathrm{Loc}(\mathcal{N})$. Then

$$\mathrm{lvl}(\mathbf{x}.\mathbf{q}) = \mathbf{s}^\top \mathbf{x} + \mathrm{lvl}(\mathbf{q}) \le \mathrm{lvl}(\mathbf{q}) \le \mathrm{cap}(\mathcal{N}),$$

where the last inequality is due to Lemma 5.18(iii). Since the choice of $\mathbf{x}.\mathbf{q}$ is arbitrary, the inequality above implies that any level greater than $\mathrm{cap}(\mathcal{N})$ is unstoppable, proving the first part of the lemma.

For the second part of the lemma, note that:

$$\begin{aligned}
\mathrm{Stop}(\mathcal{N}) =& \mathbb{N} \cap \{\mathbf{s}^\top \mathbf{x} + \mathrm{lvl}(\mathbf{q}) \mid \mathbf{x} \leq \mathbf{0} \text{ and } \mathbf{q} \in \mathrm{Loc}(\mathcal{N})\} \\
\supseteq& \mathbb{N} \cap \{\mathbf{s}^\top \mathbf{x} + \mathrm{cap}(\mathcal{N}) \mid \mathbf{x} \leq \mathbf{0}\} \qquad \text{(by Lemma 5.18(v))}. \\
=& \mathbb{N} \cap (\mathrm{cap}(\mathcal{N}) + \{\mathbf{s}^\top \mathbf{x} \mid \mathbf{x} \leq \mathbf{0}\}) \\
=& \mathbb{N} \cap (\mathrm{cap}(\mathcal{N}) + \{0, -1, -2, \ldots\}) \\
=& \{0, \ldots, \mathrm{cap}(\mathcal{N})\},
\end{aligned}$$

where the second to last equality uses the hypothesis that $\mathbf{s}$ has a coordinate equal to 1. □

Remark. The condition that $\mathbf{s}$ has a coordinate equal to 1 is not necessary for $\mathrm{Stop}(\mathcal{N})$ to be equal to $\{0, 1, \ldots, \mathrm{cap}(\mathcal{N})\}$; as can be seen from the following example.

Example 5.24. Let G be the digraph with vertex set $\{v_1, v_2\}$, and with three edges directed from v_1 to v_2, and two edges directed from v_2 to v_1. Consider the network $\mathcal{N}$ on G with the alphabet, state space, and state transition of the processor $\mathcal{P}_v$ given by

$$A_v := \{v\}, \qquad Q_v := \{0, 1, \ldots, \mathrm{indeg}(v) - 1\}, \qquad T_v(i) := i + 1 \pmod{\mathrm{indeg}(v)}.$$

For each $v \in V$, fix a total order $e_0^v, \ldots, e_{\mathrm{outdeg}(v)-1}^v$ on the outgoing edges of v.

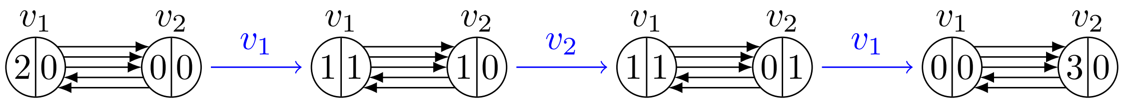

Figure 5.3: A three-step legal execution in the abelian network in Example 5.24. In the figure, the left part of a vertex records the number of letters waiting to be processed, and the right part records the state of the processor.

The message-passing function of $\mathcal{N}$ is given by:

$$T_{e_j^{v_1}}(i, v_1) := \begin{cases} v_2 & \text{if } i = j = 0; \text{ or if } i = 1 \text{ and } j \in \{1,2\}; \\ \epsilon & \text{otherwise.} \end{cases};$$

$$T_{e_j^{v_2}}(i, v_2) := \begin{cases} v_1 & \text{if } i \in \{1,2\} \text{ and } j = i-1; \\ \epsilon & \text{otherwise.} \end{cases}.$$

See Figure 5.3 for an illustration of this process.

The production matrix and the exchange rate vector of $\mathcal{N}$ are given by

$$P = \begin{bmatrix} 0 & \frac{2}{3} \\ \frac{3}{2} & 0 \end{bmatrix}, \qquad \mathbf{s} = \begin{bmatrix} 3 \\ 2 \end{bmatrix},$$

respectively. The levels of the states of $\mathcal{N}$ are given by:

$$\mathrm{lvl}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}\right) = 0, \quad \mathrm{lvl}\left(\begin{bmatrix} 0 \\ 1 \end{bmatrix}\right) = 2, \quad \mathrm{lvl}\left(\begin{bmatrix} 0 \\ 2 \end{bmatrix}\right) = 1,$$
$$\mathrm{lvl}\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}\right) = 1, \quad \mathrm{lvl}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) = 3, \quad \mathrm{lvl}\left(\begin{bmatrix} 1 \\ 2 \end{bmatrix}\right) = 2.$$

The capacity of $\mathcal{N}$ is then equal to 3, and the set of stoppable levels is given by:

$$\mathrm{Stop}(\mathcal{N}) = \{0, 1, 2, 3\}. \qquad \triangle$$

We now state the main result of this subsection, which is a refinement of Theorem 4.21 for critical networks.

Recall that the torsion group $\mathrm{Tor}(\mathcal{N})$ (Definition 4.18) acts on the set of invertible recurrent components $\overline{\mathrm{Rec}}(\mathcal{N})^{\times}$ (Definition 4.19) using the action described in Definition 4.20. Recall the definition of free and transitive actions from §4.3. Let $\mathbb{Z}_0^A := \{\mathbf{z} \in \mathbb{Z}^A \mid \mathbf{s}^\top \mathbf{z} = 0\}$, and let $\phi : \mathbb{N}^A \to \mathrm{End}(\overline{\mathrm{Rec}}(\mathcal{N}))$ be the monoid homomorphism from Definition 4.17.

Theorem 5.25. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. Then*

(i) *The map $\phi : \mathbb{N}^A \to \mathrm{End}(\overline{\mathrm{Rec}}(\mathcal{N}))$ induces an isomorphism of abelian groups*

$$\mathrm{Tor}(\mathcal{N}) \simeq \mathbb{Z}_0^A/(I-P)K.$$

(ii) $\overline{\mathrm{Rec}}(\mathcal{N})^{\times} = \displaystyle\bigsqcup_{m \in \mathbb{N}\setminus \mathrm{Stop}(\mathcal{N})} \overline{\mathrm{Rec}}(\mathcal{N}, m).$

(iii) *For any $m \in \mathbb{N} \setminus \mathrm{Stop}(\mathcal{N})$, the action of the torsion group*

$$\mathrm{Tor}(\mathcal{N}) \times \overline{\mathrm{Rec}}(\mathcal{N}, m) \to \overline{\mathrm{Rec}}(\mathcal{N}, m)$$

is free and transitive.

We remark that Theorem 1.2, stated in the introduction, is a direct corollary of Theorem 5.25(iii).

As an application of Theorem 5.25, we compute $(|\overline{\mathrm{Rec}}(\mathcal{N}, m)|)_{m \geq \mathrm{cap}(\mathcal{N})}$ for any height-arrow network $\mathcal{N}$. This generalizes [60, Theorem 1], which computes $|\overline{\mathrm{Rec}}(\mathcal{N}, \mathrm{cap}(\mathcal{N}))|$ for a sinkless rotor network $\mathcal{N}$.

Example 5.26. Let $\mathcal{N}$ be a locally irreducible sinkless height-arrow network (Example 3.13) on a strongly connected digraph G. By Theorem 5.25(i), the torsion group of $\mathcal{N}$ is isomorphic to

$$\mathrm{Tor}(\mathcal{N}) \simeq \mathbb{Z}_0^V/((D_G - A_G)\mathbb{Z}^V),$$

where D_G is the outdegree matrix of G, A_G is the adjacency matrix of G, and $\mathbb{Z}_0^V = \{\mathbf{z} \in \mathbb{Z}^V \mid \mathbf{1}^\top \mathbf{z} = 0\}$. By [34, Theorem 2.10], the cardinality of $\mathrm{Tor}(\mathcal{N})$ is then equal to the *Pham index*,

$$\mathrm{Pham}(G) := \gcd_{v \in V}\{t(G, v)\},$$

where $t(G, v)$ is the number of spanning trees of G oriented toward v. By Theorem 5.25(iii), this is also the number of recurrent components of level m, where m is any integer greater than $\mathrm{cap}(\mathcal{N})$. $\triangle$

We now build toward the proof of Theorem 5.25, and we start with a technical lemma.

Recall the definition of the relation $\dashrightarrow\!\dashleftarrow$ and $\rightarrow\!\leftarrow$ (Definition 4.6). Also recall that $\overline{\mathbf{x}.\mathbf{q}}$ denotes the component of the trajectory digraph (Definition 4.7) that contains the configuration $\mathbf{x}.\mathbf{q}$.

Lemma 5.27. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. For any $\mathbf{x}, \mathbf{x}' \in \mathbb{Z}^A$ and $\mathbf{q}, \mathbf{q}' \in \mathrm{Loc}(\mathcal{N})$,*

(i) *If $\mathrm{lvl}(\mathbf{x}.\mathbf{q}) = \mathrm{lvl}(\mathbf{x}'.\mathbf{q}')$, then there exist $\mathbf{n}, \mathbf{n}' \in \mathbb{N}^A$ such that $(\mathbf{x} + \mathbf{n}).\mathbf{q} \dashrightarrow (\mathbf{x}' + \mathbf{n}').\mathbf{q}'$ and $\mathbf{s}^\top \mathbf{n} = \mathbf{s}^\top \mathbf{n}'$.*

(ii) *If $\mathbf{x}.\mathbf{q} \dashrightarrow\!\dashleftarrow \mathbf{x}'.\mathbf{q}'$ and $\mathbf{x}.\mathbf{q}$ is a recurrent configuration, then $\mathbf{x}'.\mathbf{q}' \longrightarrow \mathbf{x}.\mathbf{q}$.*

(iii) *If $\mathrm{lvl}(\mathbf{x}.\mathbf{q}) \in \mathbb{N} \setminus \mathrm{Stop}(\mathcal{N})$, then $\mathbf{x}.\mathbf{q}$ does not halt.*

(iv) *The component* $\overline{\mathbf{x}.\mathbf{q}}$ *is a recurrent component if and only if* $\mathbf{x}.\mathbf{q}$ *does not halt.*

Proof. (i) By the local irreducibility of $\mathcal{N}$, there exist $w \in A^*$ and $\mathbf{x}'' \in \mathbb{Z}^A$ such that $\mathbf{x}.\mathbf{q} \overset{w}{\dashrightarrow} \mathbf{x}''.\mathbf{q}'$. By Lemma 5.18(ii), we then have $\mathrm{lvl}(\mathbf{x}''.\mathbf{q}') = \mathrm{lvl}(\mathbf{x}.\mathbf{q}) = \mathrm{lvl}(\mathbf{x}'.\mathbf{q}')$. In particular, we have $\mathbf{s}^\top(\mathbf{x}' - \mathbf{x}'') = 0$. Let $\mathbf{n}$ and $\mathbf{n}'$ be the positive and the negative part of $\mathbf{x}' - \mathbf{x}''$, respectively. It follows that $(\mathbf{x}+\mathbf{n}).\mathbf{q} \overset{w}{\dashrightarrow} (\mathbf{x}'+\mathbf{n}').\mathbf{q}'$ and $\mathbf{s}^\top\mathbf{n} = \mathbf{s}^\top\mathbf{n}'$.

(ii) Because $\mathbf{x}.\mathbf{q} \dashrightarrow\dashleftarrow \mathbf{x}'.\mathbf{q}'$, there exist $w_1, w_2 \in A^*$ and a configuration $\mathbf{y}.\mathbf{p}$ such that $\mathbf{x}.\mathbf{q} \overset{w_1}{\dashrightarrow} \mathbf{y}.\mathbf{p}$ and $\mathbf{x}'.\mathbf{q}' \overset{w_2}{\dashrightarrow} \mathbf{y}.\mathbf{p}$. Also note that by Lemma 5.3(iii) there is $w \in A^*$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}.\mathbf{q}$ and $|w| = \mathbf{r}$.

Let k be a positive number such that $k|w| \geq |w_2|$, and let l be a positive number such that $l|w| \geq k|w| + |w_1| - |w_2|$. (Note that k and l exist because $\mathbf{r} \geq \mathbf{1}$.) Write $w' := w^l \setminus (k|w| + |w_1| - |w_2|)$. We have

$$\begin{tikzcd}
\mathbf{x}.\mathbf{q} \arrow[rr, "w^l"] \arrow[dr, dashed, "w_1"] & & \mathbf{x}.\mathbf{q} \\
\mathbf{x}'.\mathbf{q}' \arrow[r, dashed, "w_2"] & \mathbf{y}.\mathbf{p} \arrow[r, dashed, "w^k \setminus |w_2|"] & \pi_{w^k}(\mathbf{x}'.\mathbf{q}') \arrow[u, "w'"']
\end{tikzcd},$$

where the solid arrow $\xrightarrow{w'}$ is due to the removal lemma (Lemma 4.2). Now note that since $\mathbf{q}'$ is locally recurrent, we have by Lemma 3.9 that $\pi_{w^k}(\mathbf{x}'.\mathbf{q}') = \pi_{k\mathbf{r}}(\mathbf{x}'.\mathbf{q}') = \mathbf{x}'.\mathbf{q}'$. Hence we conclude that $\mathbf{x}'.\mathbf{q}' \xrightarrow{w'} \mathbf{x}.\mathbf{q}$, as desired.

(iii) Let $\mathbf{y}.\mathbf{p}$ be any configuration such that $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{y}.\mathbf{p}$. Since $\mathbf{q}$ is locally recurrent, the state $\mathbf{p}$ is also locally recurrent by Lemma 3.5(i). By Lemma 5.18(ii) we then have $\mathrm{lvl}(\mathbf{y}.\mathbf{p}) = \mathrm{lvl}(\mathbf{x}.\mathbf{q})$. Since $\mathrm{lvl}(\mathbf{x}.\mathbf{q}) \in \mathbb{N} \setminus \mathrm{Stop}(\mathcal{N})$, it then follows that $\mathbf{y}.\mathbf{p}$ is not a stable configuration. Since the choice of $\mathbf{y}.\mathbf{p}$ is arbitrary, we then conclude that $\mathbf{x}.\mathbf{q}$ does not halt.

(iv) Proof of only if direction: Suppose to the contrary that $\mathbf{x}.\mathbf{q}$ halts. Without loss of generality, we can assume that $\mathbf{x}.\mathbf{q}$ is a stable configuration (by replacing $\mathbf{x}.\mathbf{q}$ with its stabilization if necessary).

By Lemma 5.19, the component $\overline{\mathbf{x}.\mathbf{q}}$ contains a recurrent configuration $\mathbf{y}.\mathbf{p}$. Since $\mathbf{x}.\mathbf{q} \to\leftarrow \mathbf{y}.\mathbf{p}$ and $\mathbf{y}.\mathbf{p}$ is recurrent, we have $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{y}.\mathbf{p}$. Since $\mathbf{x}.\mathbf{q}$ is stable, we then have $\mathbf{x}.\mathbf{q} = \mathbf{y}.\mathbf{p}$. Hence $\mathbf{x}.\mathbf{q}$ is both stable and recurrent, which contradicts the definition of recurrence.

Proof of if direction: Because $\mathbf{x}.\mathbf{q}$ does not halt, the component $\overline{\mathbf{x}.\mathbf{q}}$ contains a legal execution of the form:

$$\mathbf{y}_0.\mathbf{p} \xrightarrow{w_1} \mathbf{y}_1.\mathbf{p} \xrightarrow{w_2} \mathbf{y}_2.\mathbf{p} \xrightarrow{w_3} \cdots ,$$

for some $\mathbf{p} \in Q$, $\mathbf{y}_i \in \mathbb{Z}^A$, and nonempty words $w_{i+1} \in A^*$ $(i \geq 0)$. Note that for all $i \geq 0$ we have

$$\mathbf{s}^\top \mathbf{y}_i = \mathbf{s}^\top \mathbf{y}_0, \qquad \text{and} \qquad \mathbf{y}_i(a) \geq \min(\mathbf{y}_0(a), 0) \quad \forall a \in A,$$

by Lemma 3.9 and Lemma 3.3(iii), respectively. This implies that the set $\{\mathbf{y}_i \mid i \geq 0\}$ is finite. By the pigeonhole principle, there exist $j \in \mathbb{N}$ and $k \geq 1$ such that $\mathbf{y}_j = \mathbf{y}_{j+k}$.

Write $w := w_{j+1} \cdots w_k$ and $\mathbf{y} := \mathbf{y}_j = \mathbf{y}_{j+k}$. It follows that w is a nonempty word and $\mathbf{y}.\mathbf{p} \xrightarrow{w} \mathbf{y}.\mathbf{p}$. By Lemma 5.3(ii) the configuration $\mathbf{y}.\mathbf{p}$ is recurrent, and then by Lemma 5.19 the component $\overline{\mathbf{x}.\mathbf{q}} = \overline{\mathbf{y}.\mathbf{p}}$ is a recurrent component. $\square$

We now prove Theorem 5.25.

Proof of Theorem 5.25. (i) By Theorem 4.21(iii), it suffices to show that $\mathbb{Z}_0^A/(I-P)K$ is the torsion subgroup of $\mathbb{Z}^A/(I-P)K$.

By definition of $\mathbb{Z}_0^A$, the group $(I-P)K$ is a subgroup of $\mathbb{Z}_0^A$. Since K is a subgroup of $\mathbb{Z}^A$ of finite index (Lemma 3.7(i)) and P is strongly connected, it follows from the Perron-Frobenius theorem (Lemma 3.10(v)) that the $\mathbb{R}$-span of $(I-P)K$ has dimension $|A|-1$. Since the $\mathbb{R}$-span of $\mathbb{Z}_0^A$ also has dimension $|A|-1$, we conclude that the quotient group $\mathbb{Z}_0^A/(I-P)K$ is finite.

Since $\gcd_{a\in A}\mathbf{s}(a)=1$, there exists $\mathbf{s}'\in\mathbb{Z}^A$ such that $\mathbf{s}^\top\mathbf{s}'=1$. Then

$$\frac{\mathbb{Z}^A}{(I-P)K}=\frac{\mathbb{Z}_0^A}{(I-P)K}\oplus\mathbb{Z}\mathbf{s}'\simeq\frac{\mathbb{Z}_0^A}{(I-P)K}\oplus\mathbb{Z},$$

and it follows that $\tau(\mathcal{K}(\mathcal{N}))=\mathbb{Z}_0^A/(I-P)K$, as desired.

(ii) Proof of the $\supseteq$ direction: Let $\mathcal{C}$ be any recurrent component with level in $\mathbb{N}\setminus\mathrm{Stop}(\mathcal{N})$. By part (i) and Definition 4.19, it suffices to show that, for any $\mathbf{n},\mathbf{n}'\in\mathbb{N}^A$ such that $\mathbf{n}-\mathbf{n}'\in\mathbb{Z}_0^A$, there exists a recurrent component $\mathcal{C}'$ such that $\phi(\mathbf{n})(\mathcal{C})=\phi(\mathbf{n}')(\mathcal{C}')$.

By Lemma 5.19, the recurrent component $\mathcal{C}$ contains a recurrent configuration $\mathbf{x}.\mathbf{q}$. In particular, $\mathbf{q}$ is locally recurrent by Lemma 5.4(i). Write $\mathbf{x}':=\mathbf{x}+\mathbf{n}-\mathbf{n}'$. Since $\mathbf{n}-\mathbf{n}'\in\mathbb{Z}_0^A$, it follows that $\mathrm{lvl}(\mathbf{x}'.\mathbf{q})=\mathrm{lvl}(\mathbf{x}.\mathbf{q})$. In particular, we have $\mathrm{lvl}(\mathbf{x}'.\mathbf{q})\in\mathbb{N}\setminus\mathrm{Stop}(\mathcal{N})$.

By Lemma 5.27(iii), we then have $\mathbf{x}'.\mathbf{q}$ is a nonhalting configuration. By Lemma 5.27(iv), we then have $\overline{\mathbf{x}'.\mathbf{q}}$ is a recurrent component. The claim now follows by taking $\mathcal{C}':=\overline{\mathbf{x}'.\mathbf{q}}$.

Proof of the $\subseteq$ direction: Let $\mathbf{x}.\mathbf{q}$ be a recurrent configuration such that $\overline{\mathbf{x}.\mathbf{q}}\in\overline{\mathrm{Rec}}(\mathcal{N})^\times$. It follows from Lemma 5.4(ii) and Lemma 5.18(iii) that $\mathrm{lvl}(\mathbf{x}.\mathbf{q})\geq 0$.

Suppose to the contrary that $\mathrm{lvl}(\mathbf{x}.\mathbf{q})$ is in $\mathrm{Stop}(\mathcal{N})$. Then there exist $\mathbf{x}'\leq\mathbf{0}$ and $\mathbf{q}'\in\mathrm{Loc}(\mathcal{N})$ such that $\mathrm{lvl}(\mathbf{x}.\mathbf{q})=\mathrm{lvl}(\mathbf{x}'.\mathbf{q}')$. By Lemma 5.27(i), there exist $\mathbf{n},\mathbf{n}'\in\mathbb{N}^A$ such that $(\mathbf{x}+\mathbf{n}).\mathbf{q}\dashrightarrow(\mathbf{x}'+\mathbf{n}').\mathbf{q}'$ and $\mathbf{n}-\mathbf{n}'\in\mathbb{Z}_0^A$.

Since $\overline{\mathbf{x}.\mathbf{q}}$ is an invertible recurrent component and $\mathbf{n}-\mathbf{n}' \in \mathbb{Z}_0^A$, by part (i) and Definition 4.19 there exists a recurrent configuration $\mathbf{y}.\mathbf{p}$ such that $\phi(\mathbf{n})(\overline{\mathbf{x}.\mathbf{q}}) = \phi(\mathbf{n}')(\overline{\mathbf{y}.\mathbf{p}})$. Then

$$\begin{aligned}
&\phi(\mathbf{n})(\overline{\mathbf{x}.\mathbf{q}}) = \phi(\mathbf{n}')(\overline{\mathbf{y}.\mathbf{p}}) \quad \text{and} \quad (\mathbf{x}+\mathbf{n}).\mathbf{q} \dashrightarrow (\mathbf{x}'+\mathbf{n}').\mathbf{q}' \\
\implies \quad & (\mathbf{y}+\mathbf{n}').\mathbf{p} \dashrightarrow\!\!\dashleftarrow (\mathbf{x}'+\mathbf{n}').\mathbf{q}' \\
\implies \quad & \mathbf{y}.\mathbf{p} \dashrightarrow\!\!\dashleftarrow \mathbf{x}'.\mathbf{q}' \qquad \text{(by Lemma 3.3(i))} \\
\implies \quad & \mathbf{x}'.\mathbf{q}' \longrightarrow \mathbf{y}.\mathbf{p} \qquad \text{(by Lemma 5.27(ii))} \\
\implies \quad & \mathbf{x}'.\mathbf{q}' = \mathbf{y}.\mathbf{p} \qquad (\text{since } \mathbf{x}' \leq \mathbf{0}).
\end{aligned}$$

In particular we have $\mathbf{x}'.\mathbf{q}'$ is a recurrent configuration. However, this contradicts the assumption that $\mathbf{x}'.\mathbf{q}'$ is stable, and the proof is complete.

(iii) It follows from part (i) that the action of $\mathrm{Tor}(\mathcal{N})$ preserves the level of invertible recurrent component it acts on. By part (ii), it then follows that the group $\mathrm{Tor}(\mathcal{N})$ acts on $\overline{\mathrm{Rec}}(\mathcal{N}, m)$ for all $m \in \mathbb{N} \setminus \mathrm{Stop}(\mathcal{N})$. The freeness of the action follows from Theorem 4.21.

We now prove the transitivity of the action. Let $m \in \mathbb{N} \setminus \mathrm{Stop}(\mathcal{N})$. We first show that $\overline{\mathrm{Rec}}(\mathcal{N}, m)$ is nonempty. Let $\mathbf{q} \in \mathrm{Loc}(\mathcal{N})$, and let $\mathbf{x} \in \mathbb{Z}^A$ such that $\mathbf{s}^\top \mathbf{x} = m - \mathrm{lvl}(\mathbf{q})$ (Note that $\mathbf{x}$ exists because $\gcd_{a \in A} \mathbf{s}(a) = 1$). It follows that $\mathbf{x}.\mathbf{q}$ is a configuration with level $m \in \mathbb{N} \setminus \mathrm{Stop}(\mathcal{N})$. By Lemma 5.27(iii), $\mathbf{x}.\mathbf{q}$ is a nonhalting configuration. By Lemma 5.27(iv), the component $\overline{\mathbf{x}.\mathbf{q}}$ is a recurrent component. Hence $\overline{\mathrm{Rec}}(\mathcal{N}, m)$ is nonempty.

Let $\overline{\mathbf{x}'.\mathbf{q}'}$ be any recurrent component with level m. By Lemma 5.19 we can assume that $\mathbf{x}'.\mathbf{q}'$ is a recurrent configuration. In particular, $\mathbf{q}'$ is locally recurrent by Lemma 5.4(i). By Lemma 5.27(i) there exist $\mathbf{n}, \mathbf{n}' \in \mathbb{N}^A$ such that $(\mathbf{x}+\mathbf{n}).\mathbf{q} \dashrightarrow\!\!\dashleftarrow (\mathbf{x}'+\mathbf{n}').\mathbf{q}'$ and $\mathbf{n}-\mathbf{n}' \in \mathbb{Z}_0^A$. By Lemma 5.4(iii) both $\overline{(\mathbf{x}+\mathbf{n}).\mathbf{q}}$ and $\overline{(\mathbf{x}'+\mathbf{n}').\mathbf{q}'}$ are recurrent components. By Proposition 4.9, we then conclude that $\overline{(\mathbf{x}+\mathbf{n}).\mathbf{q}} = \overline{(\mathbf{x}'+\mathbf{n}').\mathbf{q}'}$. Now note that

$$\phi(\mathbf{n})(\overline{\mathbf{x}.\mathbf{q}}) = \overline{(\mathbf{x}+\mathbf{n}).\mathbf{q}} = \overline{(\mathbf{x}'+\mathbf{n}').\mathbf{q}'} = \phi(\mathbf{n}')(\overline{\mathbf{x}'.\mathbf{q}'}).$$

Since the choice of $\overline{\mathbf{x}'.\mathbf{q}'}$ is arbitrary, we conclude that the action is transitive, as desired. $\square$

CHAPTER 6

CRITICAL NETWORKS: DYNAMICS

In this chapter we study the dynamics of critical networks in more detail, with a focus on the activity and the legal executions of a configuration.

6.1 Activity as a component invariant

In this section we show that the activity of a configuration (as defined below) is a component invariant for a large family of update rules that includes the parallel update.

Definition 6.1 (Update rule). Let $\mathcal{N}$ be an abelian network. An *update rule* of $\mathcal{N}$ is an assignment of a word $u(\mathbf{x}.\mathbf{q}) \in A^*$ to each configuration $\mathbf{x}.\mathbf{q}$ such that $u(\mathbf{x}.\mathbf{q})$ is a legal execution for $\mathbf{x}.\mathbf{q}$. △

Described in words, an update rule tells the network how to process any given input configuration.

We refer to the word $u(\mathbf{x}.\mathbf{q})$ assigned to $\mathbf{x}.\mathbf{q}$ as the *update word* for $\mathbf{x}.\mathbf{q}$. The *update function* $U : \mathbb{Z}^A \times Q \to \mathbb{Z}^A \times Q$ is the function that maps a configuration $\mathbf{x}.\mathbf{q}$ to its updated configuration $\pi_{u(\mathbf{x}.\mathbf{q})}(\mathbf{x}.\mathbf{q})$. In order to simplify the notation, we use u instead of $u(\mathbf{x}.\mathbf{q})$ to denote the update word for $\mathbf{x}.\mathbf{q}$. For any $i \geq 1$, we use u_i to denote the update word for $U^{i-1}(\mathbf{x}.\mathbf{q})$. The words u' and $(u'_i)_{i\geq 1}$ for the configuration $\mathbf{x}'.\mathbf{q}'$ are defined similarly.

Recall that, for any $w \in A^*$, we denote by $|w|$ the vector in $\mathbb{N}^A$ that counts the number of occurrences of each letter in w.

Definition 6.2 (Activity vector). Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. The *activity vector* of a configuration $\mathbf{x}.\mathbf{q}$ w.r.t. a given update rule u is

$$\mathrm{act}_u(\mathbf{x}.\mathbf{q}) = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} |u_i|. \qquad \triangle$$

Described in words, the activity vector records the average number of times a letter is processed when $\mathbf{x}.\mathbf{q}$ is the input configuration.

Note that the limit in Definition 6.2 exists and is finite. This is because the sequence $\mathbf{x}.\mathbf{q}, U(\mathbf{x}.\mathbf{q}), U^2(\mathbf{x}.\mathbf{q}), \ldots$ is eventually periodic (as $\{U^i(\mathbf{x}.\mathbf{q})\}_{i\geq 0}$ is finite by criticality).

We are mainly interested in update rules that satisfy these two properties:

(H1) For any configuration $\mathbf{x}.\mathbf{q}$ such that $\mathbf{x} \in \mathbb{N}^A \setminus \{\mathbf{0}\}$, the update word u for $\mathbf{x}.\mathbf{q}$ is a nonempty word.

(H2) For any $a \in A$ and any configurations $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{a} \mathbf{x}'.\mathbf{q}'$, the update words u for $\mathbf{x}.\mathbf{q}$ and u' for $\mathbf{x}'.\mathbf{q}'$ satisfy $|u| \leq |a| + |u'|$.

The following are several examples of update rules on the sinkless sandpile network (Example 3.12) that satisfy (H1) and (H2).

Example 6.3 (Parallel update [13, 10]). The *parallel update* on the sinkless sandpile network is the rule where every unstable vertex (i.e. $v \in V$ such that $\mathbf{x}(v) + \mathbf{q}(v) \geq \mathrm{outdeg}(v)$) of the input configuration is fired once (i.e. sends one chip along every outgoing edge). Described formally, the update word u for $\mathbf{x}.\mathbf{q}$ is a word that satisfies

$$|u|(v) = \min\{\mathbf{x}(v), \mathrm{outdeg}(v)\} \qquad (v \in V).$$

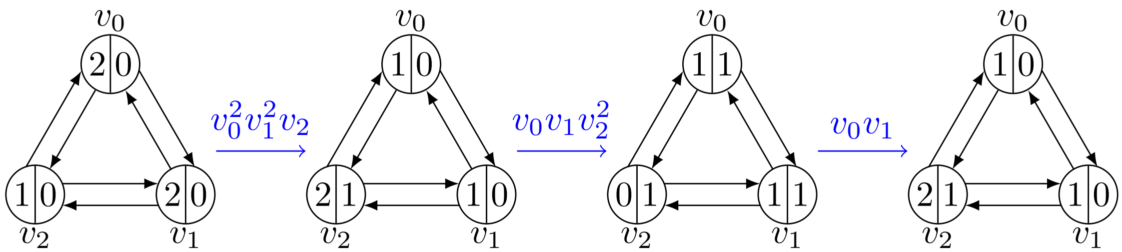

Figure 6.1: A three-step parallel update in the sinkless sandpile network on the bidirected cycle C_3. In the figure, the left part of a vertex records the number of letters waiting to be processed, and the right part records the state of the processor. Note that these configurations has activity $(1, 1, 1)^\top$, as the last two steps of this update form a periodic two-step update where every letter is fired twice.

See Figure 6.1 for an illustration of this update rule.

The parallel update satisfies (H1) by definition, and satisfies (H2) by the following computation. Let $\mathbf{d} \in \mathbb{Z}^V$ be given by $\mathbf{d}(v) := \text{outdeg}(v)$ $(v \in V)$. Then for any $v \in V$ and any configuration $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{v} \mathbf{x}'.\mathbf{q}'$,

$$\begin{aligned} |v| + |u'| =& |v| + \min\{\mathbf{x}', \mathbf{d}\} = |v| + \min\{\mathbf{x} + P|v| - |v|, \mathbf{d}\} \\ \geq& |v| + \min\{\mathbf{x} - |v|, \mathbf{d}\} \geq \min\{\mathbf{x}, \mathbf{d}\} \\ =& |u|. \end{aligned}$$

We remark that a variant of the parallel update rule where a vertex is being fired until it is stable (i.e., $|u|(v) = \mathbf{x}(v)$ for all $v \in V$) also satisfies (H1) and (H2). △

Example 6.4 (Sequential update). Fix a total order $v_0, \ldots, v_{n-1}$ on the vertices of G. The *sequential update* on the sinkless sandpile network is the rule where the vertices $v_0, \ldots, v_{n-1}$ are checked in this order, and each of them is fired once during the checking process if it is found to be unstable. Described formally, the update

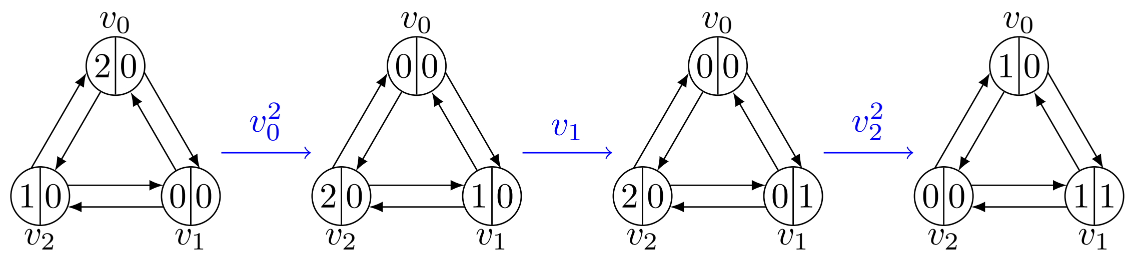

Figure 6.2: A breakdown of one-step sequential update in the sinkless sandpile network on the bidirected cycle C_3. Note that vertex v_2 is fired (i.e., sending chips to its neighbor) even though it is initially stable (i.e., has less chips than its outgoing edge).

word $u = v_0^{k_0} v_1^{k_1} \ldots v_{n-1}^{k_{n-1}}$ for $\mathbf{x}.\mathbf{q}$ satisfies:

$$k_i := \min\{\mathbf{x}_{i-1}(v), \operatorname{outdeg}(v)\} \qquad (i \in \{0, \ldots, n-1\}),$$

where $\mathbf{x}_i.\mathbf{q}_i$ is the configuration $\pi_{k_0|v_0|+\ldots+k_i|v_i|}(\mathbf{x}.\mathbf{q})$. See Figure 6.2 for an illustration of this update rule.

The sequential update satisfies (H1) by definition, and satisfies (H2) by a computation similar to Example 6.3.

Unlike the parallel update, here a vertex can potentially be fired even if the vertex is stable in the input configuration. This is because the vertex might acquire additional chips from other vertices that are checked before it; see Figure 6.2.

We remark that a mix of the parallel update and the sequential update on a partition $V_0 \cup \ldots \cup V_{k-1}$ of V (i.e., check $V_0, \ldots V_{k-1}$ in that order, and then apply the parallel update on V_i when it is being checked) also satisfies (H1) and (H2). $\triangle$

Example 6.5 (Savings update). Fix a nonempty subset $S \subseteq V$. The *savings update* works as follow:

- If there exists an unstable vertex in $V \setminus S$, then apply the parallel update on

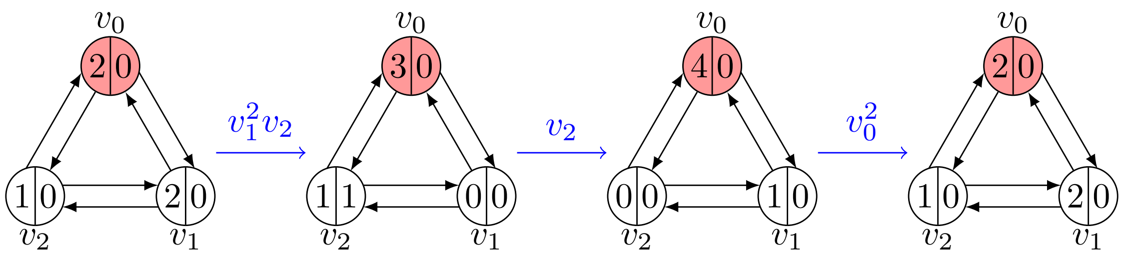

Figure 6.3: A three-step savings update in the sinkless sandpile network on the bidirected cycle C_3, with v_0 as the distinguished vertex. Note v_0 is not fired in the first step even though it is unstable. Also note that these configurations has activity $(\frac{2}{3}, \frac{2}{3}, \frac{2}{3})^\top$, as every letter is fired twice in this (periodic) three-step update.

$V \setminus S$.

- Otherwise, apply the parallel update on S.

Described in words, the vertices in S are acting as saving accounts that are used only when all other accounts are running out of funds. See Figure 6.3 for an illustration of this update rule.

Unlike the parallel and sequential updates, here it is possible for a vertex in S to not fire even if it is unstable (i.e., when there exists another unstable vertex in $V \setminus S$), as can be seen from Figure 6.3.

The savings update rule satisfies (H1) by definition, and satisfies (H2) when $S = \{v\}$ by the following argument: Let $v \in V$ and let $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ be configurations such that $\mathbf{x}.\mathbf{q} \xrightarrow{v} \mathbf{x}'.\mathbf{q}'$. There are three possible scenarios:

- All vertices are stable in $\mathbf{x}.\mathbf{q}$. In this scenario, no vertices are fired during the update of $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$, and (H2) is vacuously true.
- $V \setminus \{v\}$ is unstable in $\mathbf{x}.\mathbf{q}$. In this scenario, (H2) can be verified by the same

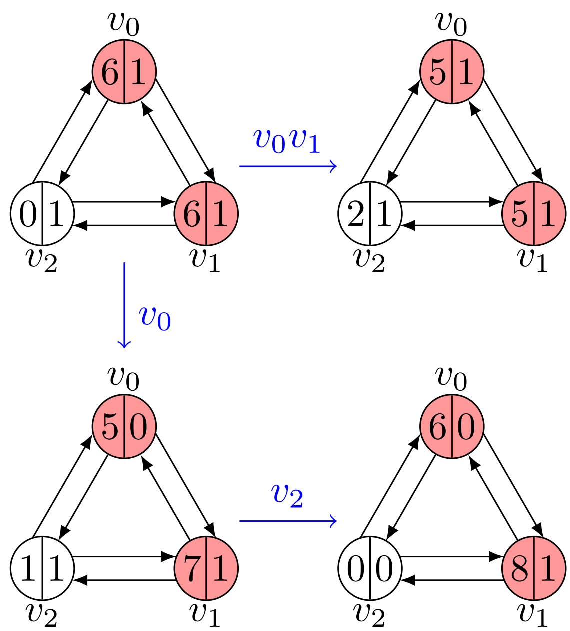

Figure 6.4: The horizontal arrows are savings updates in the sinkless sandpile network on the bidirected cycle C_3, with $S = \{v_0, v_1\}$. The update word u for the top-left configuration is v_0v_1, and the update word u' for the bottom-left configuration is v_2. The bottom-left configuration can be reached from the top-left configuration by executing the letter v_0. Note that $|u| = (1,1,0)^\top$ and $|v_0| + |u'| = (1,0,1)^\top$, so the inequality in (H2) is not satisfied.

computation in Example 6.3.

- $V \setminus \{v\}$ is stable, and v is unstable in $\mathbf{x}.\mathbf{q}$. In this scenario, the vertex v is fired during the update of $\mathbf{x}.\mathbf{q}$. Now note that, by the savings update rule, either v is fired during the update of $\mathbf{x}'.\mathbf{q}'$, or v is alredy fired during the transition from $\mathbf{x}.\mathbf{q}$ to $\mathbf{x}'.\mathbf{q}'$. In either case, the inequality in (H2) holds.

We would like to warn the reader that (H2) is not satisfied when $|S| \geq 2$; see Figure 6.4. $\triangle$

We remark that changing the update rule will usually result in changing the activity vector; see Example 6.1 and Example 6.3.

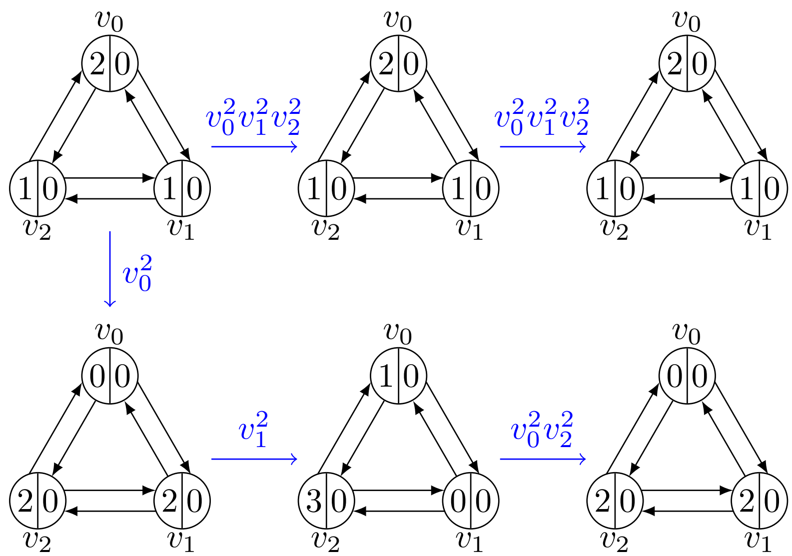

Figure 6.5: The horizontal arrows are update rules in the sinkless sandpile network on the bidirected cycle C_3. The update word u for the top-left configuration is $v_0^2v_1^2v_2^2$, the update word u' for the bottom-left configuration is v_1^2, and the update word for the bottom-middle configuration is $v_0^2v_2^2$. The bottom-left configuration can be reached from the top-left configuration by executing the letter v_0^2, and yet the former has activity $(1,1,1)^\top$ while the latter has activity $(2,2,2)^\top$. Note that $|u| = (2,2,2)^\top$ and $|v_0^2| + |u'| = (2,2,0)^\top$, so (H2) is not satisfied.

We now present the main result of this section. Recall the definition of the relation $\to\leftarrow$ from Definition 4.6.

Proposition 6.6. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical abelian network. If the given update rule u on $\mathcal{N}$ satisfies* (H1) *and* (H2)*, then $\mathbf{x}.\mathbf{q} \to\leftarrow \mathbf{x}.\mathbf{q}'$ implies $\mathrm{act}_u(\mathbf{x}.\mathbf{q}) = \mathrm{act}_u(\mathbf{x}'.\mathbf{q}')$.*

Note that the conclusion of Proposition 6.6 can fail when the hypotheses are not satisfied; see Figure 6.5.

We now build towards the proof of Proposition 6.6. We start with the following lemma that extends the conclusion in (H2) from letters to words.

Lemma 6.7. *Let $\mathcal{N}$ be an abelian network. If the given update rule on $\mathcal{N}$ satisfies (H2), then for any $w \in A^*$ and any $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ such that $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$, we have*

$$|u| \leq |w| + |u'|.$$

Proof. Write $w = a_1 \dots a_\ell$. Let $\mathbf{x}_j.\mathbf{q}_j := \pi_{a_1 \dots a_j}(\mathbf{x}.\mathbf{q})$ ($j \in \{0, \dots, \ell\}$), and let w_{j+1} be the update word for $\mathbf{x}_j.\mathbf{q}_j$. Then by (H2),

$$\begin{aligned} |u| = |w_1| \leq |a_1| + |w_2| \leq |a_1| + |a_2| + |w_3| \leq \dots \\ \leq |a_1| + \dots + |a_\ell| + |w_{\ell+1}| = |w| + |u'|. \end{aligned}$$

This proves the lemma. $\square$

We will use the following technical lemma in the proof of Proposition 6.6. Recall the definition of $w \setminus \mathbf{n}$ ($w \in A^*, \mathbf{n} \in \mathbb{N}^A$) from Definition 4.1.

Lemma 6.8. *Let $\mathcal{N}$ be an abelian network, and with a given update rule that satisfies (H2). Let $w \in A^*$ and let $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ be configurations such that $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$. Then we have the following commutative diagram:*

$$\begin{tikzcd} \mathbf{x}.\mathbf{q} \arrow[r, "u_1"] \arrow[d, "w_0"] & U(\mathbf{x}.\mathbf{q}) \arrow[r, "u_2"] \arrow[d, "w_1"] & U^2(\mathbf{x}.\mathbf{q}) \arrow[r, "u_3"] \arrow[d, "w_2"] & \cdots \\ \mathbf{x}'.\mathbf{q}' \arrow[r, "u_1'"] & U(\mathbf{x}'.\mathbf{q}') \arrow[r, "u_2'"] & U^2(\mathbf{x}'.\mathbf{q}') \arrow[r, "u_3'"] & \cdots \end{tikzcd},$$

where w_i is given by:

$$w_i := \begin{cases} w & \text{if } i = 0; \\ w_{i-1}u_i' \setminus |u_i| & \text{if } i \geq 1. \end{cases}$$

Proof. It suffices to show that $U^i(\mathbf{x}.\mathbf{q}) \xrightarrow{w_i} U^i(\mathbf{x}'.\mathbf{q}')$ for all $i \geq 0$. We will prove this claim by induction on i. The base case $i = 0$ holds since $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$

by assumption. Now assume that $U^i(\mathbf{x}.\mathbf{q}) \xrightarrow{w_i} U^i(\mathbf{x}'.\mathbf{q}')$. By Lemma 6.7, we have $|u_{i+1}| \leq |w_i| + |u'_{i+1}|$. By the removal lemma (Lemma 4.2), we then have $U^{i+1}(\mathbf{x}.\mathbf{q}) \xrightarrow{w_{i+1}} U^{i+1}(\mathbf{x}'.\mathbf{q}')$, as desired. □

We now present the proof of Proposition 6.6.

Proof of Proposition 6.6. Let $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}'.\mathbf{q}'$ be any two configurations in the same component of the trajectory digraph of $\mathcal{N}$. Note that the infinite sequence

$$\mathbf{x}'.\mathbf{q}' \xrightarrow{u'_1} U(\mathbf{x}'.\mathbf{q}') \xrightarrow{u'_2} U^2(\mathbf{x}'.\mathbf{q}') \xrightarrow{u'_3} \dots \tag{6.1}$$

is eventually periodic since the set $\{U^i(\mathbf{x}'.\mathbf{q}') \mid i \geq 0\}$ is finite (as $\mathcal{N}$ is a critical network). Also note that $\mathbf{x}'.\mathbf{q}'$ and $U^i(\mathbf{x}'.\mathbf{q}')$ have the same activity vector by Definition 6.2. Hence (by replacing $\mathbf{x}'.\mathbf{q}'$ with $U^i(\mathbf{x}'.\mathbf{q}')$ for sufficiently large i if necessary) we can without loss of generality assume that the sequence in (6.1) is periodic.

Note that by (H1), we have either $\mathbf{x}' \leq \mathbf{0}$ or the update word u'_0 for $\mathbf{x}'.\mathbf{q}'$ is nonempty. In the former scenario, we have $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$ by Definition 4.6 (since the empty word is the only legal execution for $\mathbf{x}'.\mathbf{q}'$). In the latter scenario, we have $\mathbf{x}'.\mathbf{q}'$ is a recurrent configuration by Lemma 5.3(ii) (as a consequence of (6.1) being a periodic sequence). The recurrence of $\mathbf{x}'.\mathbf{q}'$ then implies that $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$ by Definition 5.2. In both scenarios, we have $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$.

We now apply Lemma 6.8 to $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$, and let $w_0, w_1, w_2, \ldots \in A^*$ be words from Lemma 6.8. Note that, for any $i \geq 1$, we have $|u_i| \leq |w_{i-1}| + |u'_i|$ by Lemma 6.7. This implies that, for any $i \geq 1$

$$|w_i| = |w_{i-1}u'_i \setminus |u_i|| = |w_{i-1}| + |u'_i| - |u_i|.$$

Hence, for any $n \geq 0$,

$$\begin{aligned}\sum_{i=1}^{n}|u_i| =& \sum_{i=1}^{n}(|w_{i-1}| + |u'_i| - |w_i|) \qquad \text{(by Lemma 6.8)}\\ =&|w_0| - |w_n| + \sum_{i=1}^{n}|u'_i|\\ \leq&|w_0| + \sum_{i=1}^{n}|u'_i|.\end{aligned}$$

Since the equation above holds for all $n \geq 0$, it then follows from Definition 6.2 that $\mathrm{act}_u(\mathbf{x}.\mathbf{q}) \leq \mathrm{act}_u(\mathbf{x}'.\mathbf{q}')$. By symmetry we then conclude that $\mathrm{act}_u(\mathbf{x}.\mathbf{q}) = \mathrm{act}_u(\mathbf{x}'.\mathbf{q}')$, as desired. □

6.2 Near uniqueness of legal executions

In this section we estimate the proportion of any letter in a legal execution, up to an additive constant.

We assume throughout this section that $\mathcal{N}$ is a finite, locally irreducible, and strongly connected critical network.

Let $p(\cdot,\cdot)$ be the $A \times A$ matrix given by

$$p(a,b) := \frac{\mathbf{s}(b)}{\mathbf{s}(a)}P(b,a),$$

where P is the production matrix (Definition 3.8) and $\mathbf{s}$ is the exchange rate vector of $\mathcal{N}$ (i.e. the unique positive integer vector for which $\mathbf{s}P = \mathbf{s}$ and $\gcd_{a\in A}\mathbf{s}(a) = 1$). Since P is a nonnegative matrix, and $\mathbf{s}P = \mathbf{s}$ by the assumption that $\mathcal{N}$ is critical, it follows that $p(\cdot,\cdot)$ is a probability transition matrix for a Markov chain on A.

For letters $a, b, z \in A$, let $\mathfrak{G}_z(b,a)$ be the expected number of visits to a strictly before hitting z, when the Markov chain starts at b. Let $\mathbf{v}_{a,z} \in \mathbb{R}^A_{\geq 0}$ be the vector

$$\mathbf{v}_{a,z}(\cdot) := \frac{\mathbf{s}(\cdot)}{\mathbf{s}(a)} \, \mathfrak{G}_z(\cdot, a).$$

In the special case that $\mathcal{N}$ is a sandpile or rotor network on an undirected graph, the above quantities have familiar interpretations in terms of random walk and electrical networks (see, for example, [56, chapter 2]): $\mathbf{s} = \mathbf{1}$ and p is the transition matrix for simple random walk, $\mathfrak{G}_z$ is the *Green function* for the random walk absorbed at z, $\mathbf{v}_{a,z}$ is the *voltage function* for the unit current flow from a to z, and the quantity $\frac{\mathbf{v}_{a,z}(a)}{\deg(a)}$ is the *effective resistance* $R_{\text{eff}}(a, z)$ between a and z.

Recall that $\mathbf{M}_w(\mathbf{q}) \in \mathbb{N}^A$ is the vector that records numbers of letters generated by executing w at state $\mathbf{q}$. For any $\mathbf{q}, \mathbf{q}' \in \text{Loc}(\mathcal{N})$, let $\text{diff}_{a,z}(\mathbf{q}, \mathbf{q}')$ between $\mathbf{q}$ and $\mathbf{q}'$ be given by

$$\text{diff}_{a,z}(\mathbf{q}, \mathbf{q}') := \mathbf{v}_{a,z}^\top (P|w| - \mathbf{M}_w(\mathbf{q})),$$

where w is any (not necessarily legal) execution that sends $\mathbf{q}$ to $\mathbf{q}'$. Note that w exists because $\mathcal{N}$ is locally irreducible and finite, and also note that $P|w| - \mathbf{M}_w(\mathbf{q})$ does not depend on the choice of w by Lemma 3.9.

We now present the main result of this section. Recall that $\mathbf{r}$ is the period vector of $\mathcal{N}$ (Definition 5.1), and $\mathbf{1}$ is the vector $(1, \dots, 1)^\top$. For any $\mathbf{n} \in \mathbb{N}^A$, we denote by $||\mathbf{n}||$ the sum $\sum_{a \in A} \mathbf{n}(a)$.

Theorem 6.9. *Let $\mathcal{N}$ be a finite, locally irreducible, strongly connected, and critical network, and let $\mathbf{q}, \mathbf{q}' \in \text{Loc}(\mathcal{N})$. Then for any legal execution w that sends $\mathbf{x}.\mathbf{q}$ to $\mathbf{x}'.\mathbf{q}'$,*

$$-\frac{||\mathbf{c}||}{||\mathbf{r}||}\mathbf{r}(a) - \mathbf{r}(a) < |w|(a) - \frac{\ell}{||\mathbf{r}||}\mathbf{r}(a) < \mathbf{r}(a) + \mathbf{c}(a) \qquad \forall a \in A.$$

where ℓ is the length of the execution w, and $\mathbf{c} \in \mathbb{R}^A$ is the vector given by

$$\mathbf{c}(a) := \max_{z \in A} \left(\mathbf{v}_{a,z}^\top (\mathbf{x} - \mathbf{x}') + \text{diff}_{a,z}(\mathbf{q}', \mathbf{q}) \right).$$

Note that the vector $\mathbf{c}$ can be bounded from above by a positive vector that depends only on $\mathbf{x}.\mathbf{q}$ (as $\mathbf{x}'$ is lower bounded by the negative part of $\mathbf{x}$ by Lemma 3.3(iii), and there are only finitely many choices for $\mathbf{q}'$). In particular, Theorem 6.9 implies that all legal executions of a configuration of a given length are equal up to permutation and an additive constant that does not depend on the executions.

We now build towards the proof of Theorem 6.9. We will start with the following lemma relating $|w|(a)$ and $|w|(z)$.

Lemma 6.10. *Let $\mathcal{N}$ be a finite,locally irreducible, strongly connected, and critical network, and let $\mathbf{q}, \mathbf{q}' \in \mathrm{Loc}(\mathcal{N})$. Then for any $a, z \in A$ and any legal execution w sending $\mathbf{x}.\mathbf{q}$ to $\mathbf{x}'.\mathbf{q}'$, we have:*

$$|w|(a) = \mathbf{v}_{a,z}^{\top}(\mathbf{x} - \mathbf{x}') + \mathrm{diff}_{a,z}(\mathbf{q}', \mathbf{q}) + \frac{\mathbf{r}(a)}{\mathbf{r}(z)}|w|(z).$$

Proof. If $a = z$, then the lemma follows immediately from the fact that $\mathbf{v}_{a,a}$ is the zero vector. Therefore, it suffices to prove the lemma for when a is not equal to z.

By a direct computation, we have

$$(I - P^{\top})\mathbf{v}_{a,z}(b) = \begin{cases} 1 & \text{if } b = a; \\ -\frac{\mathbf{r}(a)}{\mathbf{r}(z)} & \text{if } b = z; \\ 0 & \text{if } b \in A \setminus \{a, z\}. \end{cases} \tag{6.2}$$

In particular, this implies that

$$\mathbf{v}_{a,z}^{\top}(I - P)|w| = |w|(a) - \frac{\mathbf{r}(a)}{\mathbf{r}(z)}|w|(z). \tag{6.3}$$

Let w' be a word such that $t_{w'}(\mathbf{q}') = \mathbf{q}$. Note that we have $\pi_{ww'}(\mathbf{x}.\mathbf{q}) =$

$\pi_{w'}(\mathbf{x}'.\mathbf{q}') = (\mathbf{x}' + \mathbf{M}_{w'}(\mathbf{q}') - |w'|).\mathbf{q}$. By Lemma 3.9, we then have

$$(I - P)(|w| + |w'|) = \mathbf{x} - (\mathbf{x}' + \mathbf{M}_{w'}(\mathbf{q}') - |w'|),$$

which is equivalent to

$$(I - P)|w| = (\mathbf{x} - \mathbf{x}') + (P|w'| - \mathbf{M}_{w'}(\mathbf{q}')).$$

Together with (6.3), this implies that:

$$|w|(a) - \frac{\mathbf{r}(a)}{\mathbf{r}(z)}|w|(z) = \mathbf{v}_{a,b}^{\top}(\mathbf{x} - \mathbf{x}') + \mathrm{diff}_{a,b}(\mathbf{q}', \mathbf{q}).$$

This proves the lemma. □

Remark. Lemma 6.10 implies the following inequality from [43, Proposition 4.8]: If $\mathcal{N}$ is the sandpile network on an undirected graph and $\mathbf{x}.\mathbf{q}$ is a configuration such that $\mathbf{x} \geq \mathbf{0}$ and $\mathbf{q} = (0, \dots, 0)^{\top}$, then any legal execution w for $\mathbf{x}.\mathbf{q}$ that does not contain the letter z satisfies

$$\ell \leq 2|E|\,||\mathbf{x}||\max_{a \in A} R_{\mathrm{eff}}(a, z), \tag{6.4}$$

where ℓ is the length of the execution w. Indeed, this is because for all $a \in A$:

$$\begin{aligned}
|w|(a) =& \mathbf{v}_{a,z}^{\top}(\mathbf{x} - \mathbf{x}') + \mathrm{diff}_{a,z}(\mathbf{q}', \mathbf{q}) \qquad \text{(by Lemma 6.10)} \\
\leq& \mathbf{v}_{a,z}^{\top}(\mathbf{x} - \mathbf{x}') \qquad (\text{since } \mathrm{diff}_{a,z}(\mathbf{q}', \mathbf{q}) \leq 0 \text{ if } \mathbf{q} = (0, \dots, 0)) \\
\leq& \mathbf{v}_{a,z}^{\top}\mathbf{x} \qquad (\text{since } \mathbf{x}' \geq \mathbf{0} \text{ if } w \text{ is legal}) \\
\leq& \mathbf{v}_{a,z}(a)||\mathbf{x}|| \qquad (\text{since } \mathbf{v}_{a,z}(b) \leq \mathbf{v}_{a,z}(a) \text{ for all } b \in A) \\
=& \deg(a) R_{\mathrm{eff}}(a, z)||\mathbf{x}||.
\end{aligned}$$

(6.4) now follows by summing the inequality $|w|(a) \leq \deg(a)R_{\mathrm{eff}}(a, z)||\mathbf{x}||$ over all letters in A.

We now present the proof of Theorem 6.9.

Proof of Theorem 6.9. Let k be the largest nonnegative integer such that $k\mathbf{r} \leq |w|$. Write $w' := w \setminus k\mathbf{r}$. Note that w' is a legal execution for $\mathbf{x}.\mathbf{q}$ by the removal lemma (Lemma 4.2). Also note that, by the maximality assumption, there exists $z \in A$ such that $|w'|(z) < \mathbf{r}(z)$. By Lemma 6.10, we then have for all $a \in A$:

$$|w'|(a) < \mathbf{v}_{a,z}^{\top}(\mathbf{x} - \mathbf{x}') + \mathrm{diff}_{a,z}(\mathbf{q}', \mathbf{q}) + \mathbf{r}(a) \leq \mathbf{c}(a) + \mathbf{r}(a).$$

This implies that, for all $a \in A$,

$$k\mathbf{r}(a) \leq |w|(a) < (k+1)\mathbf{r}(a) + \mathbf{c}(a). \tag{6.5}$$

Summing (6.5) over all letters in A, we get:

$$k||\mathbf{r}| \leq \ell < (k+1)||\mathbf{r}|| + ||\mathbf{c}||,$$

which implies that

$$\frac{\ell}{||\mathbf{r}||} - \frac{||\mathbf{c}||}{||\mathbf{r}||} - 1 < k \leq \frac{\ell}{||\mathbf{r}||}. \tag{6.6}$$

The proposition now follows from (6.5) and (6.6). $\square$

CHAPTER 7

SANDPILE NETWORKS

In this chapter we apply the theory of critical networks developed in previous sections to count (1) the number of recurrent states of sandpile networks with a sink and (2) the number of recurrent components of sinkless sandpile networks. The main results of this chapter apply only to the case when the underlying digraph is Eulerian (i.e. when $\operatorname{outdeg}(v) = \operatorname{indeg}(v)$ for all $v \in V(G)$).

7.1 Counting recurrent states

Let G be a strongly connected digraph, and let $\mathcal{N}$ be the sinkless sandpile network (Example 3.12) on G. Let s be any vertex of G, let $R := V(G) \setminus \{s\}$ and let $\mathcal{N}_R$ be the thief network (§5.2) based on $\mathcal{N}$ restricted to R. Note that $\mathcal{N}_R$ is also the sandpile network with a sink at s from Example 3.14. As we will often discuss more than on graph together in this section, we will write $V(G)$ and $E(G)$ instead of V and E to avoid confusion.

It was conjectured by Biggs [9] and was later proved by Merino López [57] that, when G is bidirected, the number of recurrent states of $\mathcal{N}_R$ is related to the number of spanning trees of G in the following manner.

An *undirected spanning tree* T of G is an undirected subgraph of G that is connected and consists of $|V(G)| - 1$ edges . Fix an arbitrary total order on the undirected edges of G. An undirected edge e of G that is not contained in T is *externally active* with respect to T if e is the smallest edge (with respect to the given total order) in the unique cycle of the graph $T \cup \{e\}$. The *external activity* $\operatorname{ext}(T)$ of T is the number of undirected edges of G that are externally active with respect to T.

Recall that the level of a state a sandpile network is the total number of chips in the state (see Example 5.15).

Theorem 7.1 ([57, Theorem 3.6]). *Let G be a strongly connected bidirected graph, let s be any vertex of G, and let $\mathcal{N}_R$ be the sandpile network of G with a sink at s. Then, for any $m \geq 0$, the number of $\mathcal{N}_R$-recurrent states with level m is equal to*

$$\sum_{i=1}^{\mathrm{outdeg}(s)} c_{m+i-\frac{|E(G)|}{2}}, \tag{7.1}$$

where c_k is the number of undirected spanning trees of G with external activity k. $\square$

Remark. Note that the quantity (7.1) is slightly different from the corresponding quantity in [57, Theorem 3.6] due to the different convention for defining a sink; see Example 3.14 for the detail.

An immediate consequence of Theorem 7.1 is that, when G is bidirected, the following equality holds:

$$\frac{1}{y^{|E(G)|/2-1} + y^{|E(G)|/2-2} + \ldots + y^{|E(G)|/2-\mathrm{outdeg}(s)}} \sum_{m \geq 0} a_m y^m = \sum_{T} y^{\mathrm{ext}(T)}, \tag{7.2}$$

where a_m is the number of $\mathcal{N}_R$-recurrent states with level m, and T is summed over all undirected spanning trees of G. The expression at the right side of (7.2) is the *Tutte polynomial* [68] of G evaluated at $x = 1$, which does not depend on the choice of the sink s. This implies that the left side of (7.2) also does not depend on the choice of the sink s when G is bidirected, and the same phenomen was proved by Perrot and Pham [59] to hold for Eulerian digraphs (but fails to hold for general digraphs). This prompted them to ask if Theorem 7.1 can also be extended to Eulerian digraphs. We will answer their question positively in the remainder of this section.

A *directed path* P of G of length ℓ is a sequence $e_1 \dots e_\ell$ such that for $i \in \{1, \dots, \ell - 1\}$ the target vertex of e_i is the source vertex of e_{i+1}.

Definition 7.2 (Arborescences). Let G be a strongly connected digraph. An *arborescence* T of G rooted at $s \in V(G)$ is a subgraph of G such that

(i) T contains $|V(G)| - 1$ edges;

(ii) $\mathrm{indeg}(s) = 0$ and $\mathrm{indeg}(v) = 1$ for any $v \in V(G) \setminus \{s\}$; and

(iii) For any $v \in V(G)$ there exists a unique directed path from s to v in T. $\triangle$

Fix a total order $<$ on the directed edges of G. For any two distinct edge-disjoint directed paths P_1 and P_2, we write $P_1 < P_2$ if the smallest edge in $E(P_1) \sqcup E(P_2)$ (with respect to $<$) is contained in P_1.

The following definition of external activity is due to by Björner, Korte, and Lovász [11], which was used to define a single variable generalization of the Tutte polynomial for digraphs called the greedoid polynomial.

Definition 7.3 (External activity). Let T be an arborescence of a strongly connected digraph G rooted at s. For any (directed) edge $e \in E(G) \setminus E(T)$, there are exactly two edge-disjoint directed paths P_1 and P_2 that share the same starting vertex and ending vertex in $T \sqcup \{e\}$. Let P_1 be the path that contains e. We say that e is *externally active* with respect to T if $P_1 < P_2$. The *external activity* $\mathrm{ext}(T)$ of T is the number of edges in G that are externally active with respect to T. $\triangle$

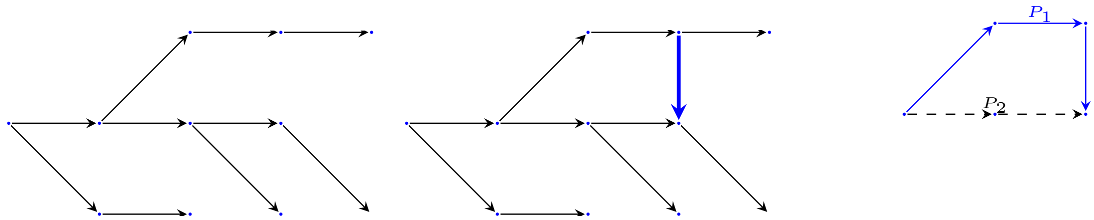

Figure 7.1: (a) An arborescence T. (b) An arborescence T with an extra edge e. (c) The path P_1 that contains e (undashed) and the path P_2 that does not contain e (dashed).

See Figure 7.1 for an illustration describing Definition 7.3.

We are now ready to state the main result of this section.

Theorem 7.4. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}_R$ be the sandpile network of G with a sink at s. Then, for any $m \geq 0$, the number of $\mathcal{N}_R$-recurrent states with level m is equal to*

$$\sum_{i=1}^{\operatorname{outdeg}(s)} c_{m+i},$$

where c_k is the number of arborescences of G rooted at s with external activity k. □

Remark. We remark that Theorem 7.1 is a special case of Theorem 7.4. Note that the extra term $-\frac{|E(G)|}{2}$ in the right side of (7.1) is because, for every pair of directed edges (v_1, v_2) of the bidirected graph G, at least one of the directed edges will be externally active, and both directed edges will be externally active if and only if the undirected edge $\{v_1, v_2\}$ is externally active.

We now build toward the proof of Theorem 7.4. Our starting point is an algorithm that maps an $\mathcal{N}_R$-recurrent state $\mathbf{q}$ with level m and with zero chips at s to an arborescence of G with external activity $m + \operatorname{outdeg}(s)$. See Algorithm 2 for a full description of the algorithm. We remark that this algorithm is a digraph version of Cori-Le Borgne bijection [26, 5] for undirected graphs.

Input:

$\mathcal{N}_R$-recurrent state $\mathbf{q}$ with zero chips at s,

Total order on the edges of G.

Output:

Arborescence $T_{\mathbf{q}}$ of G rooted at s.

Initialization:

BV $:= \{s\}$ (burnt vertices),

BE $:= \emptyset$ (burnt edges),

$T := \emptyset$ (directed tree).

while BV $\neq V(G)$ **do**

  $e := \max\{(v_1, v_2) \in E(G) \,|\, (v_1, v_2) \notin \mathrm{BE},\ v_1 \in \mathrm{BV}, v_2 \notin \mathrm{BV}\}$,

  $v_2 :=$ the target vertex of e,

  if $\mathbf{q}(v_2) == \mathrm{outdeg}(v_2) - 1 - |\{e \in \mathrm{BE}(\mathbf{q}) \mid \mathrm{trgt}(e) = v_2\}|$ **then**

    $\mathrm{BV} \leftarrow \mathrm{BV} \cup \{v_2\}$,

    $T \leftarrow T \cup \{e\}$,

  else

    $\mathrm{BE} \leftarrow \mathrm{BE} \cup \{e\}$

  end

end

Output $T_{\mathbf{q}} := T$.

Algorithm 2: Cori-Le Borge bijection from recurrent states to arborescences.

The next proposition shows that Algorithm 2 is in fact a bijection.

Proposition 7.5. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}_R$ be the sandpile network of G with a sink at s. Then Algorithm 2 is a bijection between $\mathcal{N}_R$-recurrent states with zero chips at s to arborescences of G rooted at s. Furthermore, this bijection maps a recurrent state with level m to an arborescence with external activity $m + \mathrm{outdeg}(s)$.*

We will use the following recurrence test for $\mathcal{N}_R$ from [43] that applies to all Eulerian digraphs.

Theorem 7.6 ([43, Theorem 4.4]). *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}_R$ be the sandpile network of G with a sink at s. A state $\mathbf{q}$ of $\mathcal{N}_R$ is recurrent if and only if, for any nonempty subset $Z \subseteq V(G) \setminus \{s\}$, there exists $v \in Z$ such that $\mathbf{q}(v)$ is greater than or equal to the number of edges from Z to v.* □

Remark. Theorem 7.6 can be derived from Theorem 5.7 and the fact that the period vector $\mathbf{r}$ (Definition 5.1) of $\mathcal{N}_R$ is equal to the outdegree vector when G is an Eulerian digraph; see [17] for the detail.

Note that the recurrence test in Theorem 7.6 does not impose any condition on the number of chips at s, which gives us the following corollary.

Corollary 7.7. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}_R$ be the sandpile network of G with a sink at s. Let $\mathbf{q}_1$ and $\mathbf{q}_2$ be two states of $\mathcal{N}_R$ such that $\mathbf{q}_1(v) = \mathbf{q}_2(v)$ for any $v \in V(G) \setminus \{s\}$. Then $\mathbf{q}_1$ is $\mathcal{N}_R$-recurrent if and only if $\mathbf{q}_2$ is $\mathcal{N}_R$-recurrent.* □

For any $\mathcal{N}_R$-recurrent state $\mathbf{q}$ with zero chips at s, we denote by $T_{\mathbf{q}}$ the output of Algorithm 2, we denote by $\mathrm{BV}(\mathbf{q})$ the set of vertices that are burnt in Algorithm 2, and we denote by $\mathrm{BE}(\mathbf{q})$ the set of edges that are burnt in Algorithm 2. For any $e \in E(G)$, we denote by $\mathrm{src}(e)$ the source vertex of e and by $\mathrm{trgt}(e)$ the target vertex of e.

Lemma 7.8. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}_R$ be the sandpile network of G with a sink at s, and let $\mathbf{q}$ be an $\mathcal{N}_R$-recurrent with zero chips at s. Then:*

(i) $\mathrm{BV}(\mathbf{q}) = V(G)$*;*

(ii) $\mathbf{q}(v) = \mathrm{outdeg}(v) - 1 - |\{e \in \mathrm{BE}(\mathbf{q}) \mid \mathrm{trgt}(e) = v\}|$ *for all* $v \in V(G) \setminus \{s\}$*;*

(iii) $T_{\mathbf{q}}$ *is an arborescence of* G *rooted at* s*.*

Proof. (i) Suppose to the contrary that Algorithm 2 terminates when $\mathrm{BV}(\mathbf{q})$ is a strict subset of $V(G)$. Line 5-14 of the algorithm imply that all edges with source vertex in $\mathrm{BV}(\mathbf{q})$ and target vertex in $V(G) \setminus \mathrm{BV}(\mathbf{q})$ are burnt. Write $Z := V(G) \setminus \mathrm{BV}(\mathbf{q})$. Line 8 of the algorithm then implies that, for all $v \in Z$, the function $\mathbf{q}(v)$ is strictly less than the number of edges from Z to v. This contradicts Theorem 7.6 and the assumption that $\mathbf{q}$ is $\mathcal{N}_R$-recurrent, as desired.

(ii) Since $\mathrm{BV}(\mathbf{q}) = V(G)$ by Lemma 7.8(i), Line 8 of Algorithm 2 implies that $\mathbf{q}(v)$ is equal to $\mathrm{outdeg}(v) - 1 - |\{e \in \mathrm{BE}(\mathbf{q}) \mid \mathrm{trgt}(e) = v\}|$, as desired.

(iii) It follows from Line 5-14 of Algorithm 2 that $T_{\mathbf{q}}$ is a directed tree with $|\mathrm{BV}(\mathbf{q})| - 1$ edges and with s as the unique source vertex. Since $\mathrm{BV}(\mathbf{q}) = V(G)$ by Lemma 7.8(i), it then follows that $T_{\mathbf{q}}$ is an arborescence of G rooted at s. □

Lemma 7.9. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}_R$ be the sandpile network of G with a sink at s, and let $\mathbf{q}$ be an $\mathcal{N}_R$-*

recurrent state with zero chips at s. Then an edge $e \in E(G) \setminus E(T_{\mathbf{q}})$ is externally active with respect to $T_{\mathbf{q}}$ if and only if e is not contained in $\mathrm{BE}(\mathbf{q})$.

Proof. Let P_1 and P_2 be two edge-disjoint directed paths as in Definition 7.3. Note that e is contained in P_1 by definition. Let e' be the minimum edge in $E(P_1) \sqcup E(P_2)$. We need to show that e' is contained in P_2 if and only if e is contained in $\mathrm{BE}(\mathbf{q})$.

Suppose that e' is contained in P_2. By the minimality of e', it then follows that the source vertex of e is burnt before e' in the while loop of Algorithm 2. Again by the minimality of e', it then follows that e is evaluated before e' in the while loop of the algorithm. Since e is not contained in $T_{\mathbf{q}}$, it then follows that e is burnt when it is evaluated. This proves one direction of the claim.

Suppose that e' is contained in P_1. By the minimality of e', it then follows that all edges in P_2 are evaluated before e' in the while loop of Algorithm 2. This implies that all vertices in P_2 is burnt before e' is evaluated by the while loop. Since P_1 and P_2 share the same target vertex and e is the last edge in P_1, it then follows that e is either not evaluated or evaluated after its target vertex is burnt in the while loop. In both cases e is not burnt in the while loop. This proves the other direction of the claim. $\square$

We now give an algorithm that provides the inverse map to Algorithm 2 (which, at this point of the proof, have not been shown to be a bijection yet). See Algorithm 3 for the description of the algorithm.

For any arborescence T of G, denote by f_T the output of Algorithm 3, denote by $\mathrm{BV}(T)$ the set of vertices that are burnt in Algorithm 3, and by $\mathrm{BE}(T)$ the set

Input:

Arborescence T of G rooted at s,

Total order on the edges of G.

Output:

$\mathcal{N}_R$-recurrent state $\mathbf{q}_T$ with zero chips at s.

Initialization:

BV $:= \{s\}$ (burnt vertices),

BE $:= \emptyset$ (burnt edges).

while BV $\neq V(G)$ **do**

  $e := \max\{(v_1, v_2) \in E(G) \mid (v_1, v_2) \notin \mathrm{BE},\ v_1 \in \mathrm{BV}, v_2 \notin \mathrm{BV}\}$,

  $v_2 :=$ the target vertex of e,

  if $e \in E(T)$ **then**

    $\mathrm{BV} \leftarrow \mathrm{BV} \cup \{v_2\}$,

  else

    $\mathrm{BE} \leftarrow \mathrm{BE} \cup \{e\}$

  end

end

Output $\mathbf{q}_T$, with

$$\mathbf{q}_T(v) := \begin{cases} 0 & \text{if } v = s; \\ \mathrm{outdeg}(v) - 1 - |\{e \in \mathrm{BE} \mid \mathrm{trgt}(e) = v\}| & \text{if } v \in V(G) \setminus \{s\}. \end{cases}$$

Algorithm 3: Cori-Le Borge bijection from arborescences to reverse G-parking functions.

of edges that are burnt in Algorithm 3.

Lemma 7.10. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}_R$ be the sandpile network of G with a sink at s, and let T be an arborescence of G rooted at s. Then:*

(i) $\mathbf{q}_T$ *is an $\mathcal{N}_R$-recurrent state with zero chips at s; and*

(ii) *For any $\mathcal{N}_R$-recurrent state $\mathbf{q}$ with zero chips at s, we have $\mathbf{q}_{T_{\mathbf{q}}} = \mathbf{q}$.*

Proof. (i) It follows from Line 14 of ALgorithm 3 that $\mathbf{q}_T$ has zero chips at s. Let Z be an arbitrary nonempty subset of $V(G)\setminus\{s\}$. Since T is an arborescence of G, it follows that Algorithm 3 terminates only when all vertices are burnt. Let v be the first vertex in Z that is burnt by Algorithm 3, and let e' be the edge that causes v to be added to the set of burnt vertices. By Line 7-11 of Algorithm 3, the edge e' is an unburnt edge with v as its target vertex. Furthermore, by the minimality assumption on v, the source vertices of e' and all burnt edges with v as its target vertex are contained in $V(G) \setminus Z$. These two facts imply that

$$\{e \in \mathrm{BE}(T) \mid \mathrm{trgt}(e) = v\} \subseteq \{e \in E(G) \mid \mathrm{src}(e) \in V(G)\setminus Z, \mathrm{trgt}(e) = v\}\setminus\{e'\},$$

which in turn implies that

$$|\{e \in \mathrm{BE}(T) \mid \mathrm{trgt}(e) = v\}| \leq (\#\text{ of edges from } V(G) \setminus Z \text{ to } v) - 1. \quad (7.3)$$

We then have

$$\begin{aligned}\mathbf{q}_T(v) =& \mathrm{outdeg}(v) - 1 - |\{e \in \mathrm{BE}(T) \mid \mathrm{trgt}(e) = v\}|\\ =& \mathrm{indeg}(v) - 1 - |\{e \in \mathrm{BE}(T) \mid \mathrm{trgt}(e) = v\}| \qquad \text{(since } G \text{ is Eulerian)}\\ \geq& \text{number of edges from } Z \text{ to } v \qquad \text{(by (7.3))}.\end{aligned}$$

Since the choice of Z is arbitrary, it then follows from Theorem 7.6 that $\mathbf{q}_T$ is $\mathcal{N}_R$-recurrent.

(ii) It follows from the description of Algorithm 2 and Algorithm 3 that $\mathrm{BE}(\mathbf{q}) = \mathrm{BE}(T_{\mathbf{q}})$. It then follows from Lemma 7.8(ii) that $\mathbf{q} = \mathbf{q}_{T_{\mathbf{q}}}$. □

Proof of Proposition 7.5. It follows from Lemma 7.8(iii), Lemma 7.10(i), and Lemma 7.10(ii) that Algorithm 2 and Algorithm 3 are bijections that are inverses of each other. Furthermore, for any $\mathcal{N}_R$-recurrent state $\mathbf{q}$ with zero chips at s, we have

$$\begin{aligned}
\mathrm{ext}(T_{\mathbf{q}}) &= |E(G)| - |V(G)| + 1 - |\mathrm{BE}(\mathbf{q})| \qquad \text{(by Lemma 7.9)}\\
&= |E(G)| - |V(G)| + 1 + \sum_{v \in V(G)\setminus\{s\}} \mathbf{q}(v) - \mathrm{outdeg}(v) - 1 \qquad \text{(by Lemma 7.8(ii))}\\
&= \mathrm{lvl}(\mathbf{q}) + \mathrm{outdeg}(s).
\end{aligned}$$

The proof is now complete. □

Proof of Theorem 7.4. We have that the number of $\mathcal{N}_R$-recurrent states with level m is equal to

$$\begin{aligned}
&|\{\mathbf{q} \mid \mathbf{q} \text{ is } \mathcal{N}_R\text{-recurrent}, \ \mathrm{lvl}(\mathbf{q}) = m\}|\\
=& \sum_{j=0}^{\mathrm{outdeg}(s)-1} |\{\mathbf{q} \mid \mathbf{q} \text{ is } \mathcal{N}_R\text{-recurrent}, \ \mathrm{lvl}(\mathbf{q}) = m, \ \mathbf{q}(s) = j\}|\\
=& \sum_{j=0}^{\mathrm{outdeg}(s)-1} |\{\mathbf{q} \mid \mathbf{q} \text{ is } \mathcal{N}_R\text{-recurrent}, \ \mathrm{lvl}(\mathbf{q}) = m - j, \ \mathbf{q}(s) = 0\}| \qquad \text{(by Corollary 7.7)}\\
=& \sum_{j=0}^{\mathrm{outdeg}(s)-1} c_{m-j+\mathrm{outdeg}(s)} \qquad \text{(by Proposition 7.5)}\\
=& \sum_{i=1}^{\mathrm{outdeg}(s)} c_{m+i},
\end{aligned}$$

as desired. □

7.2 Counting recurrent components

In this section we turn to the problem of counting the number of recurrent components of sinkless sandpile networks. As we will often discuss the networks $\mathcal{N}$ and $\mathcal{N}_R$ together in this section, we will specify the networks we are referring to in the notation (e.g. using $\mathcal{N}$-recurrence instead of recurrence).

Theorem 7.11. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}$ be the sinkless sandpile network of G. Then, for any $m \geq 0$, the number of $\mathcal{N}$-recurrent components with level m is equal to*

$$|\overline{\mathrm{Rec}}(\mathcal{N}, m)| = \sum_{k \leq m} c_k,$$

where c_k is the number of arborescences of G rooted at s with external activity k.

We now build toward the proof of Theorem 7.11. Fix $m \geq 0$, we denote by $\mathcal{B} := \mathcal{B}_m$ the map given by

$$\mathcal{B} : \{\mathbf{q} \mid \mathbf{q} \text{ is } \mathcal{N}_R\text{-recurrent}, \ \mathrm{lvl}(\mathbf{q}) \leq m - \mathrm{outdeg}(s), \ \mathbf{q}(s) = 0\} \to \overline{\mathrm{Rec}}(\mathcal{N}, m)$$
$$\mathbf{q} \mapsto \overline{\mathbf{x}.\mathbf{q}},$$

where $\mathbf{x} := \mathbf{x}_{\mathbf{q}}$ is given by $\mathbf{x}(s) := m - j$ and $\mathbf{x}(v) := 0$ for all $v \in V(G) \setminus \{s\}$.

Proposition 7.12. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}$ be the sinkless sandpile network of G. Then, for any $m \geq 0$, the map $\mathcal{B}$ is a bijection between $\mathcal{N}_R$-recurrent states with level less than or equal to $m - \mathrm{outdeg}(s)$ and with zero chips at s and and $\mathcal{N}$-recurrent components with level equal to m.*

Remark. When G is not Eulerian, the map $\mathcal{B}$ is instead an n-to-one correspondence with n being the Pham index (Example 5.26) of G. The proof can be found in the paper [23] by the author.

We will now show that the image of $\mathcal{B}$ is contained in $\mathrm{Rec}(\mathcal{N}, m)$. Recall that $\mathbf{r}$ denotes the period vector (Definition 5.1) of $\mathcal{N}$. Since G is Eulerian, it follows from (5.1) that

$$\mathbf{r}(v) = \mathrm{outdeg}(v) \qquad (v \in V). \tag{7.4}$$

Write $\mathbf{h} := (I - P_R)\mathbf{r}$, , where P_R denotes the production matrix of $\mathcal{N}_R$. It follows from (7.4) that

$$\mathbf{h}(v) := \begin{cases} \mathrm{outdeg}(s) & \text{if } v = s; \\ 0 & \text{if } v \in V \setminus \{s\}. \end{cases} \tag{7.5}$$

Lemma 7.13. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}$ be the sinkless sandpile network of G. Let $m \geq 0$, let $\mathbf{q}$ be a state with level $j \leq m - \mathrm{outdeg}(s)$ and with zero chips at s, and let $\mathbf{x} \in \mathbb{N}^A$ be given by $\mathbf{x}(s) := m - j$ and $\mathbf{x}(v) := 0$ for all $v \in V \setminus \{s\}$. If $\mathbf{q}$ is an $\mathcal{N}_R$-recurrent state, then $\mathbf{x}.\mathbf{q}$ is an $\mathcal{N}$-recurrent configuration with level m.*

Proof. It follows from the definition that that $\mathbf{x}.\mathbf{q}$ is a configuration of $\mathcal{N}$ with level m, and it suffices to show that $\mathbf{x}.\mathbf{q}$ is an $\mathcal{N}$-recurrent configuration.

Since $\mathbf{q}$ is $\mathcal{N}_R$-recurrent, we have by Proposition 5.11 that $\mathbf{h}\,.\mathbf{q}$ is an $\mathcal{N}$-recurrent configuration. On the other hand, we have $\mathbf{h} \leq \mathbf{x}$ by (7.5) and the assumption that $j \leq m - \mathrm{outdeg}(s)$. Together with Lemma 5.4(iii), these two facts imply that $\mathbf{x}.\mathbf{q}$ is an $\mathcal{N}$-recurrent configuration, and the proof is complete. □

We will now show that $\mathcal{B}$ is an injective map.

Lemma 7.14. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}$ be the sinkless sandpile network of G. Then, for any $m \geq 0$, the map $\mathcal{B}$ is injective.*

Proof. Let $\mathbf{q}_1$ and $\mathbf{q}_2$ be two $\mathcal{N}_R$-recurrent states with zero chips at s. Let $\mathbf{x}_1.\mathbf{q}_1$ and $\mathbf{x}_2.\mathbf{q}_2$ be the corresponding $\mathcal{N}$-recurrent configurations given by the map $\mathcal{B}$, and suppose that $\overline{\mathbf{x}_1.\mathbf{q}_1} = \overline{\mathbf{x}_2.\mathbf{q}_2}$. It suffices to show that $\mathbf{q}_1 = \mathbf{q}_2$.

By the definition of $\mathcal{N}$-recurrence (Definition 5.2), we have $\mathbf{x}_1.\mathbf{q}_1 \xrightarrow[\mathcal{N}]{} \mathbf{x}_2.\mathbf{q}_2$. By Lemma 5.12(ii), there exists $\mathbf{x}'_1, \mathbf{x}'_2 \in \mathbb{N}^A$ with $\operatorname{supp}(\mathbf{x}'_1), \operatorname{supp}(\mathbf{x}'_2) \subseteq \{s\}$ such that

$$\mathbf{x}'_1.\mathbf{q}_1 \xrightarrow[\mathcal{N}_R]{} \mathbf{x}'_2.\mathbf{q}_2. \tag{7.6}$$

Since $\mathbf{q}_1$ is $\mathcal{N}_R$-recurrent, it follows from Theorem 5.7 (by taking $\mathbf{r}$ to be the burning vector) and (7.5) that

$$\mathbf{h}\,.\mathbf{q}_1 \xrightarrow[\mathcal{N}_R]{} \mathbf{0}.\mathbf{q}_1.$$

This implies that

$$\mathbf{x}'_1.\mathbf{q}_1 \xrightarrow[\mathcal{N}_R]{} (\mathbf{x}'_1 - \mathbf{h}).\mathbf{q}_1 \xrightarrow[\mathcal{N}_R]{} \cdots \xrightarrow[\mathcal{N}_R]{} \left(\mathbf{x}'_1 - \left\lfloor \frac{\mathbf{x}'_1(s)}{\operatorname{outdeg}(s)} \right\rfloor \mathbf{h}\right).\mathbf{q}_1. \tag{7.7}$$

Let $k_1 := \mathbf{x}'_1(s) - \lfloor \frac{\mathbf{x}'_1(s)}{\operatorname{outdeg}(s)} \rfloor$, Note that $k_1 < \operatorname{outdeg}(s)$. Let $\mathbf{q}'_1$ be the state given by $\mathbf{q}'_1(s) := k$ and $\mathbf{q}'_1(v) := \mathbf{q}_1(v)$ for any $v \in V \setminus \{s\}$. It follows from the definition that

$$\left(\mathbf{x}'_1 - \left\lfloor \frac{\mathbf{x}'_1(s)}{\operatorname{outdeg}(s)} \right\rfloor \mathbf{h}\right).\mathbf{q}_1 \xrightarrow[\mathcal{N}_R]{s^{k_1}} \mathbf{0}.\mathbf{q}'_1. \tag{7.8}$$

Combining (7.7) and (7.8), we have

$$\mathbf{x}'_1.\mathbf{q}_1 \xrightarrow[\mathcal{N}_R]{} \mathbf{0}.\mathbf{q}'_1. \tag{7.9}$$

By the same reasoning, we also have

$$\mathbf{x}'_2.\mathbf{q}_2 \xrightarrow[\mathcal{N}_R]{} \mathbf{0}.\mathbf{q}'_2. \tag{7.10}$$

On the other hand, combining (7.6) and (7.10) gives us

$$\mathbf{x}'_1.\mathbf{q}_1 \xrightarrow[\mathcal{N}_R]{} \mathbf{0}.\mathbf{q}'_2. \tag{7.11}$$

As the executions in (7.9) and (7.11) are both legal and complete, it follows from the least action principle (Corollary 4.3) that $\mathbf{q}_1' = \mathbf{q}_2'$. Together with the assumption that $\mathbf{q}_1(s) = \mathbf{q}_2(s) = 0$, this allows us to conclude that $\mathbf{q}_1 = \mathbf{q}_2$ as desired. □

We will now show that $\mathcal{B}$ is a surjective map.

Lemma 7.15. *Let G be a strongly connected Eulerian digraph, let s be any vertex of G, and let $\mathcal{N}$ be the sinkless sandpile network of G. Then, for any $m \geq 0$, the map $\mathcal{B}$ is surjective.*

Proof. Let $\mathbf{x}.\mathbf{q}$ be any $\mathcal{N}$-recurrent configuration of level m. Since $\mathcal{N}_R$ is a subcritical network (Lemma 5.8(ii)), there exists $\mathbf{q}' \in Q$ such that $\mathbf{x}.\mathbf{q} \xrightarrow[\mathcal{N}_R]{} \mathbf{0}.\mathbf{q}'$. By Lemma 5.12(ii), there exists $\mathbf{x}' \in \mathbb{N}^A$ with $\operatorname{supp}(\mathbf{x}') \subseteq \{s\}$ such that $\mathbf{x}.\mathbf{q} \xrightarrow[\mathcal{N}]{} \mathbf{x}'.\mathbf{q}'$. By Proposition 5.4(iv), this implies that $\mathbf{x}'.\mathbf{q}'$ is $\mathcal{N}$-recurrent. By Proposition 5.9, this in turn implies that $\mathbf{q}'$ is $\mathcal{N}_R$-recurrent.

Let $\mathbf{q}'' \in Q$ be given by $\mathbf{q}''(s) := 0$ and $\mathbf{q}''(v) := \mathbf{q}'(v)$ for all $v \in V \setminus \{s\}$. Since $\mathbf{q}'$ is $\mathcal{N}_R$-recurrent, it follows from Corollary 7.7 that $\mathbf{q}''$ is also $\mathcal{N}_R$-recurrent.

Let $\mathbf{x}'' \in \mathbb{N}^A$ be given by $\mathbf{x}''(s) := \mathbf{x}'(s) + \mathbf{q}'(s)$ and $\mathbf{x}''(v) := \mathbf{x}'(v)$ for any $v \in V \setminus \{s\}$. We then have

$$\mathbf{x}''.\mathbf{q}'' \xrightarrow[\mathcal{N}]{s^{\mathbf{q}'(s)}} \mathbf{x}'.\mathbf{q}'.$$

Together with the fact that $\mathbf{x}.\mathbf{q} \xrightarrow[\mathcal{N}]{} \mathbf{x}'.\mathbf{q}'$, this implies that $\mathbf{x}.\mathbf{q}$ and $\mathbf{x}''.\mathbf{q}''$ are contained in the same $\mathcal{N}$-recurrent components.

Now note that level of $\mathbf{q}''$ is given by

$$\begin{aligned} \mathrm{lvl}(\mathbf{q}'') &= \mathrm{lvl}(\mathbf{x}''.\mathbf{q}'') - \mathbf{x}''(s) \qquad \text{(since } \mathbf{s} = \mathbf{1} \text{ by Table 5.1)} \\ &= \mathrm{lvl}(\mathbf{x}.\mathbf{q}) - \mathbf{x}''(s) \qquad \text{(by Lemma 5.18(ii))} \\ &= m - \mathbf{x}''(s). \end{aligned}$$

Also note that $\mathbf{x}''(s) \geq \mathrm{outdeg}(s)$, as otherwise $s^{\mathrm{outdeg}(s)}$ is a complete execution for $\mathbf{x}''.\mathbf{q}''$, which contradicts the previous conclusion that $\mathbf{x}''.\mathbf{q}''$ is contained in an $\mathcal{N}$-recurrent component. These two facts imply that $\mathrm{lvl}(\mathbf{q}'') \leq m - \mathrm{outdeg}(s)$.

Putting everything together, we conclude that $\mathbf{q}''$ is an element of the domain of $\mathcal{B}$ that is mapped to $\overline{\mathbf{x}.\mathbf{q}}$ by $\mathcal{B}$. This proves the surjectivity of $\mathcal{B}$. $\square$

Proof of Proposition 7.12. The proposition follows from Lemma 7.13, Lemma 7.14, and Lemma 7.15. $\square$

We are now ready to present the proof of Theorem 7.11.

Proof of Theorem 7.11. The theorem follows from Proposition 7.5 and Proposition 7.12. $\square$

CHAPTER 8

ROTOR AND AGENT NETWORKS

An *abelian mobile agent network* [15, Example 3.7], or *agent network* for short, is an abelian network in which every processor $\mathcal{P}_v$ produces one letter of output for each letter of input. Formally, an agent network is an abelian network such that for all $a \in A$ and $\mathbf{q} \in Q$ we have $\mathbf{1}^\top \mathbf{M}_a(\mathbf{q}) = 1$ (Recall that $\mathbf{M}_a(\mathbf{q}) \in \mathbb{N}^A$ is the vector recording the number of letters of each type that are produced when the network in state $\mathbf{q}$ processes the letter a).

Examples of agent networks include sinkless rotor networks (Example 3.11) and inverse networks (Example 3.19), while non-examples include sinkless sandpile networks (Example 3.12) and arithmetical networks (Example 3.15).

Any agent network is a critical network. Indeed, by the definition of agent networks, for any $\mathbf{q} \in Q$ and any $w \in A^*$,

$$\mathbf{1}^\top \mathbf{M}_w(\mathbf{q}) = \sum_{a \in A} |w|(a) = \mathbf{1}^\top |w|,$$

where $|w| \in \mathbb{N}^A$ is the vector that counts the number of occurrences of each letter in w. This implies that the production matrix P satisfies

$$\mathbf{1}^\top P = \mathbf{1}. \tag{8.1}$$

By the Perron-Frobenius theorem (Lemma 3.10(ii)), the spectral radius $\lambda(P)$ is equal to 1. Hence an agent network is a critical network.

We assume throughout this chapter that the agent network we are working with is finite, locally irreducible, and strongly connected, unless stated otherwise.

Special to agent networks is the notion of rotor digraph.

Definition 8.1 (Rotor digraph). Let $\mathcal{N}$ be an agent network. For $\mathbf{q} \in \mathrm{Loc}(\mathcal{N})$, the *rotor digraph* $\varrho_{\mathbf{q}}$ is the digraph

$$V(\varrho_{\mathbf{q}}) := A, \qquad E(\varrho_{\mathbf{q}}) := \{(a, a_{\mathbf{q}}) \mid a \in A\},$$

where $a_{\mathbf{q}}$ is the letter produced when the network $\mathcal{N}$ in state $t_a^{-1}(\mathbf{q})$ processes the letter a. △

Rotor digraphs belong to a special family of digraphs called cycle-rooted forests, defined as follows. A *cycle-rooted tree* is the disjoint union of a directed tree rooted at a vertex r and an edge with source vertex r. Note that a cycle-rooted tree contains a unique directed cycle, and for every vertex v in the digraph there is a directed path from v to the cycle. A *cycle-rooted forest* is a disjoint union of cycle-rooted trees. Equivalently, a cycle-rooted forest is a digraph in which every vertex has outdegree equal to 1.

The following are two examples of rotor digraphs.

Example 8.2. Consider the sinkless rotor network (Example 3.11) on the bidirected cycle C_4.

Let $\mathbf{q} \in \Pi_{k \in \mathbb{Z}_4} \mathrm{Out}(v_k)$ be the state given by

$$\mathbf{q}(k) := (v_k, v_{k+1}) \qquad (k \in \mathbb{Z}_4).$$

See Figure 8.1 for an illustration.

On processing the letter v_k, the state $T_{v_k}^{-1}((v_k, v_{k+1})) = (v_k, v_{k-1})$ produces the letter v_{k+1}, and therefore the rotor digraph $\varrho_{\mathbf{q}}$ contains the edge (v_k, v_{k+1}). This gives us the rotor digraph $\varrho_{\mathbf{q}}$ in Figure 8.1.

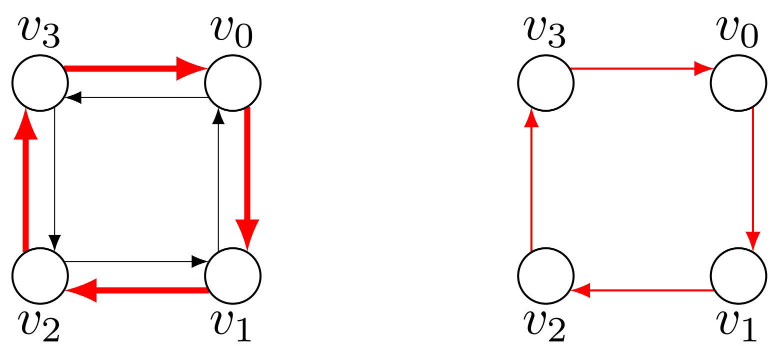

Figure 8.1: The figure on the left is the state $\mathbf{q} := ((v_k, v_{k+1}))_{k\in\mathbb{Z}_4}$ (given by the (red) thick edges) of a sinkless rotor network, and the figure on the right is the rotor digraph of $\mathbf{q}$.

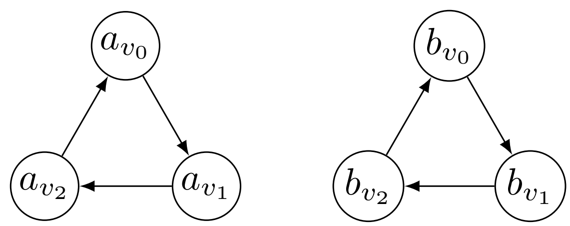

Figure 8.2: For the inverse network on the bidirected cycle C_3, the rotor digraph of the state $\mathbf{q} := (1, 1, 1)$ is a disjoint union of two directed triangles. Note that the state $\mathbf{q}' := (2, 2, 2)$ has the same rotor digraph.

By a similar reasoning, for a sinkless rotor network on an arbitrary digraph G, the rotor digraph $\varrho_{\mathbf{q}}$ of any state $\mathbf{q}$ is given by

$$V(\varrho_{\mathbf{q}}) = V(G), \qquad E(\varrho_q) = \{\mathbf{q}(v) \mid v \in V(G)\}.$$

In particular, if G is a simple digraph, then the state $\mathbf{q}$ is determined by its rotor digraph $\varrho_{\mathbf{q}}$. This is not true for arbitrary agent networks, as shown in the next example. $\triangle$

Example 8.3. Consider the inverse network (Example 3.19) on the bidirected cycle C_3 with period $m_{v_k} = 6$ for all $v_k \in V$ and with the message-passing function in Table 8.1.

Table 8.1: The message-passing function for the processor $\mathcal{P}_{v_k}$ ($k \in \mathbb{Z}_3$). The (q,α)-th entry of the table represents the letter produced when a processor in state q processes the letter α.

A_{v_k} \ Q_{v_k}	0	1	2	3	4	5
a_{v_k}	$a_{v_{k+1}}$	$a_{v_{k+1}}$	$a_{v_{k+1}}$	$a_{v_{k+1}}$	$a_{v_{k+1}}$	$b_{v_{k+1}}$
b_{v_k}	$a_{v_{k+1}}$	$b_{v_{k+1}}$	$b_{v_{k+1}}$	$b_{v_{k+1}}$	$b_{v_{k+1}}$	$b_{v_{k+1}}$

The states $\mathbf{q} := (1,1,1)$ and $\mathbf{q}' := (2,2,2)$ have the same rotor digraph, as shown in Figure 8.2. However, on processing the input $b_{v_0}b_{v_0}$,

- The network at state $\mathbf{q}$ produces $b_{v_1}a_{v_1}$ as output; while
- The network at state $\mathbf{q}'$ produces $b_{v_1}b_{v_1}$ as output.

Hence a state is not determined by its rotor digraph in this inverse network. $\triangle$

This chapter is structured as follows. In §8.1 we derive an efficient recurrence test for agent networks. In §8.2 and §8.3 we apply the methods developed in §5.2 to count the recurrent components and recurrent configurations of an agent network, respectively.

8.1 The cycle test for recurrence

In this section we present a recurrence test for agent networks that is more efficient than the burning test in §5.1.

A *directed walk* in the rotor digraph $\varrho_{\mathbf{q}}$ is a sequence $a_1, \ldots, a_{\ell+1} \in A^*$ such

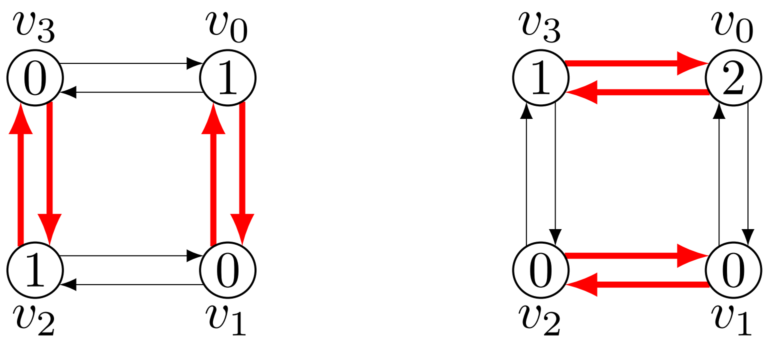

Figure 8.3: Two configurations in the sinkless rotor network on the bidirected cycle C_4. The circled number by vertex v_i indicates the number of chips $\mathbf{x}(v_i)$, and the (red) thick outgoing edge from v_i records the rotor $\mathbf{q}(v_i)$. By the cycle test, the configuration on the left is recurrent while the configuration on the right is not recurrent.

that $(a_i, a_{i+1}) \in E(\varrho_{\mathbf{q}})$ for $i \in \{1, \ldots, \ell\}$. A *directed path* in $\varrho_{\mathbf{q}}$ is a directed walk in which all a_i's are distinct except possibly for a_1 and $a_{\ell+1}$. A *directed cycle* in $\varrho_{\mathbf{q}}$ is a directed path in which $a_1 = a_{\ell+1}$.

Recall that the support of $\mathbf{x} \in \mathbb{Z}^A$ is $\operatorname{supp}(\mathbf{x}) = \{a \in A : \mathbf{x}(a) \neq 0\}$.

Theorem 8.4 (Cycle test). *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected agent network. A configuration $\mathbf{x}.\mathbf{q}$ is recurrent if and only if all these conditions are satisfied:*

(C1) *The vector $\mathbf{x}$ is nonnegative;*

(C2) *The state $\mathbf{q}$ is locally recurrent; and*

(C3) *Every directed cycle of the rotor digraph $\varrho_{\mathbf{q}}$ contains a vertex in $\operatorname{supp}(\mathbf{x})$.*

We remark that Theorem 1.4 in §1.8 is the special case of Theorem 8.4 when $\mathcal{N}$ is a sinkless rotor network (so that $\varrho_{\mathbf{q}} = \mathbf{q}$).

Theorem 8.4 answers the question posed in [17] for a characterization of recurrent configurations of agent networks.

In the case of sinkless rotor network, Theorem 8.4 implies that the recurrence of a configuration $\mathbf{x}.\mathbf{q}$ is not influenced by the choice of cyclic order for the rotor mechanism, as the rotor digraph $\rho_{\mathbf{q}}$ itself does not depend on the cyclic order (see Example 8.2). The independence from the cyclic order is a recurring phenomenon that has been observed for other invariants in the rotor network [22].

The cycle test is often much more computationally efficient than the burning test (Algorithm 1). In particular, for a sinkless rotor network on an n-vertex directed graph, conditions (C1)-(C3) can be checked in time linear in n.

The following is a corollary of Theorem 8.4 that we will use later in §8.2.

Corollary 8.5. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected agent network. Let $\mathbf{x}$ and $\mathbf{x}'$ be nonnegative vectors such that* $\operatorname{supp}(\mathbf{x}) = \operatorname{supp}(\mathbf{x}')$. *For any $\mathbf{q} \in Q$, the configuration $\mathbf{x}.\mathbf{q}$ is recurrent if and only if $\mathbf{x}'.\mathbf{q}$ is recurrent.* □

We now build toward the proof of Theorem 8.4, and we start with two technical lemmas. Recall that, for any $w \in A^*$, we denote by $|w|$ the vector in $\mathbb{N}^A$ that counts the occurrences of each letter in w.

Lemma 8.6. *Let $\mathcal{N}$ be a finite and locally irreducible agent network. Let* $\mathbf{q} \in \operatorname{Loc}(\mathcal{N})$ *and let $a_1 \dots a_{\ell+1}$ be a directed path in $\varrho_{\mathbf{q}}$. Write $w' := a_1 \dots a_\ell$ and $\mathbf{q}' := t_{a_1}^{-1} \cdots t_{a_\ell}^{-1}\mathbf{q}$, then*

$$|a_1|.\mathbf{q}' \xrightarrow{w'} |a_{\ell+1}|.\mathbf{q}.$$

Proof. We prove the claim by induction on ℓ. When $\ell = 0$, the claim is true since $\mathbf{w}'$ is the empty word, $a_1 = a_{l+1}$, and $\mathbf{q}' = \mathbf{q}$.

We now prove the claim for when $\ell \geq 1$. Write $w'' := a_2 \dots a_{\ell+1}$ and $\mathbf{q}'' := t_{a_2}^{-1} \cdots t_{a_\ell}^{-1}\mathbf{q}$. By the induction hypothesis we have $|a_2|.\mathbf{q}'' \xrightarrow{w''} |a_{\ell+1}|.\mathbf{q}$. Since a_1 is a legal execution for $|a_1|.\mathbf{q}'$, it then suffices to show that $\pi_{a_1}(|a_1|.\mathbf{q}') = |a_2|.\mathbf{q}''$.

Now note that

$$\mathbf{M}_{w'}(\mathbf{q}') = \mathbf{M}_{a_1}(\mathbf{q}') + \mathbf{M}_{w''}(\mathbf{q}'') = \mathbf{M}_{a_1}(\mathbf{q}') + |a_3| + \cdots + |a_{\ell+1}|,$$

where the last equality is due to $\pi_{w''}(|a_2|.\mathbf{q}'') = |a_{\ell+1}|.\mathbf{q}$. Also note that

$$\begin{aligned}
\mathbf{M}_{w'}(\mathbf{q}') =& \mathbf{M}_{a_1 \cdots a_\ell}(t_{a_1}^{-1} \cdots t_{a_\ell}^{-1}\mathbf{q}) \\
=& \mathbf{M}_{a_2 \cdots a_\ell a_1}(t_{a_2}^{-1} \cdots t_{a_\ell}^{-1} t_{a_1}^{-1}\mathbf{q}) \quad \text{(by the abelian property (Lemma 3.1(ii)))} \\
=& \mathbf{M}_{a_2 \cdots a_\ell}(t_{a_2}^{-1} \cdots t_{a_\ell}^{-1} t_{a_1}^{-1}\mathbf{q}) + \mathbf{M}_{a_1}(t_{a_1}^{-1}\mathbf{q}) \\
\geq& \mathbf{M}_{a_1}(t_{a_1}^{-1}\mathbf{q}) = |a_2|,
\end{aligned}$$

where the last equality is because (a_1, a_2) is an edge in $\varrho_{\mathbf{q}}$. These two equations then imply that

$$\mathbf{M}_{a_1}(\mathbf{q}') + |a_3| + \cdots + |a_{\ell+1}| \geq |a_2|. \tag{8.2}$$

Now note that $a_2 \notin \{a_3, \dots, a_{\ell+1}\}$ since $a_1 \dots a_{\ell+1}$ is a directed path in $\varrho_{\mathbf{q}}$. It then follows from (8.2) that $\mathbf{M}_{a_1}(\mathbf{q}') \geq |a_2|$. Since $\mathcal{N}$ is an agent network, we conclude that $\mathbf{M}_{a_1}(\mathbf{q}') = |a_2|$. It then follows that $\pi_{a_1}(|a_1|.\mathbf{q}') = |a_2|.\mathbf{q}''$, and the proof is complete. □

Recall that $\mathbf{r}$ denotes the period vector of $\mathcal{N}$ (Definition 5.1). Also recall the definition of $w \setminus \mathbf{n}$ ($w \in A^*, \mathbf{n} \in \mathbb{N}^A$) from Definition 4.1.

Lemma 8.7. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected agent network. Then for any $\mathbf{q} \in \mathrm{Loc}(\mathcal{N})$ and any $a \in A$ there exists a legal execution w for $|a|.\mathbf{q}$ such that $|w|(a) = \mathbf{r}(a) + 1$ and $|w| \leq \mathbf{r} + |a|$.*

Proof. Fix a letter $a \in A$. Let $w' = a_1 \cdots a_\ell$ be a word of maximum length such that w' is a legal execution for $|a|.\mathbf{q}$ and $|w'| \leq \mathbf{r}$.

Write $a' := \mathbf{M}_{a_\ell}(t_{a_1 \cdots a_{\ell-1}} \mathbf{q})$ and $w := w'a'$. It follows that w is a legal execution for $|a|.\mathbf{q}$. Note that $|w|(a') = \mathbf{r}(a') + 1$, as otherwise we would have $|w| \leq \mathbf{r}$ and that contradicts the maximality of w. Also note that $|w| = |w'| + |a'| \leq \mathbf{r} + |a'|$.

We now show that $a' = a$. Since $\mathcal{N}$ is an agent network and w' is a legal execution for $|a|.\mathbf{q}$, we have $\mathbf{M}_{a_i}(t_{a_1 \cdots a_{i-1}} \mathbf{q}) = |a_{i+1}|$ for any $i \in \{1, \ldots, \ell - 1\}$. Hence

$$\mathbf{M}_{w'}(\mathbf{q}) = \sum_{i=1}^{\ell} \mathbf{M}_{a_i}(t_{a_1 \cdots a_{i-1}} \mathbf{q}) = \sum_{i=1}^{\ell-1} |a_{i+1}| + |a'| = |w| - |a_1|.$$

Then

$$|a_1| = |w| - \mathbf{M}_{w'}(\mathbf{q}) \geq |w| - \mathbf{M}_{\mathbf{r}}(\mathbf{q}) = |w| - \mathbf{r}, \tag{8.3}$$

where the inequality is due to $|w'| \leq \mathbf{r}$ and the monotonicity property (Lemma 3.1(i)), and the last equality is due to $\mathbf{q} \in \mathrm{Loc}(\mathcal{N})$. Since $|w|(a') = \mathbf{r}(a') + 1$, (8.3) implies that $|a_1|(a') \geq 1$, and hence we have $a_1 = a'$.

Now note that $a_1 = a$ because $w = a_1 \cdots a_\ell$ is a legal execution for $|a|.\mathbf{q}$. Hence $a' = a_1 = a$, and it then follows that w satisfies the property in the lemma. □

We now present the proof of Theorem 8.4. Recall that a word $w \in A^*$ is called a-tight if $|w| \leq \mathbf{r}$ and $|w|(a) = \mathbf{r}(a)$.

Proof of Theorem 8.4. Proof of if direction: Since $\mathbf{q}$ is locally recurrent by (C2), by Lemma 5.5 it suffices to show that for each $a \in A$ there exists an a-tight legal execution w for $\mathbf{x}.\mathbf{q}$.

Fix a letter $a \in A$. Let $a_1, \ldots, a_{\ell+1}$ be a directed path of minimum length in $\varrho_{\mathbf{q}}$ such that $a_1 = a$ and $a_{\ell+1} \in \operatorname{supp}(\mathbf{x})$. Note that such a directed path exists by (C3). Write $w' := a_1 \cdots a_\ell$ and $\mathbf{q}' := t_{a_1}^{-1} \cdots t_{a_\ell}^{-1}\mathbf{q}$. Note that $|a|.\mathbf{q}' \xrightarrow{w'} |a_{\ell+1}|.\mathbf{q}$ by Lemma 8.6. Also note that $|w'|(a) = 1$ and $|w'| \leq \mathbf{1} \leq \mathbf{r}$ by the minimality assumption.

By Lemma 8.7, there exists an legal execution w'' for $|a|.\mathbf{q}'$ such that $|w''|(a) = \mathbf{r}(a) + 1$ and $|w''| \leq \mathbf{r} + |a|$. Write $w := w'' \setminus |w'|$. By the removal lemma (Lemma 4.2), w is a legal execution for $|a_{\ell+1}|.\mathbf{q}$. Since $\mathbf{x} \in \mathbb{N}^A$ (by (C1)) and $a_{\ell+1} \in \operatorname{supp}(\mathbf{x})$, by Lemma 3.3(ii) we conclude that w is a legal execution for $\mathbf{x}.\mathbf{q}$.

We now show that w is a-tight. Note that

$$
\begin{aligned}
|w| &= \max(|w''|, |w'|) - |w'| \\
&\leq \max(|w''|, |w'|) - |a| \quad (\text{since } |w'|(a) = 1) && (8.4) \\
&\leq \mathbf{r} + |a| - |a| \quad (\text{since } |w''| \leq \mathbf{r} + |a| \text{ and } |w'| \leq \mathbf{r}) && (8.5) \\
&= \mathbf{r}.
\end{aligned}
$$

Also note that we have equality for the a-th coordinate in (8.4) (because $|w'|(a) = 1$) and (8.5) (because $|w''|(a) = \mathbf{r}(a) + 1$). Hence we conclude that $|w| \leq \mathbf{r}$ and $|w|(a) = \mathbf{r}(a)$, i.e., the word w is a-tight. This completes the proof.

Proof of only if direction: It suffices to show that (C3) holds, as (C1) and (C2) follow from Lemma 5.4. Let $a_1, \ldots, a_{\ell+1}$ be any directed cycle in $\varrho_{\mathbf{q}}$. Note that $a_{\ell+1} = a_1$ by assumption. We need to show that $\{a_1, \ldots, a_\ell\} \cap \operatorname{supp}(\mathbf{x})$ is nonempty.

By Theorem 5.6, there exists a legal execution w for $\mathbf{x}.\mathbf{q}$ such that $|w| = \mathbf{r}$ and $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}.\mathbf{q}$. Write $\mathbf{n} := \mathbf{r} - \sum_{i=1}^{\ell} |a_i|$ and $w' := w \setminus \mathbf{n}$. Note that $\mathbf{n}$ is a nonnegative vector (because $\mathbf{r} \geq \mathbf{1}$ and $a_1, \ldots, a_\ell$ are distinct), and w' is a permutation of the word $a_1 \dots a_\ell$. Write $\mathbf{x}'.\mathbf{q}' := \pi_{\mathbf{n}}(\mathbf{x}.\mathbf{q})$. By the removal lemma, we have $\mathbf{x}'.\mathbf{q}' \xrightarrow{w'} \mathbf{x}.\mathbf{q}$.

Since w' is legal for $\mathbf{x}'.\mathbf{q}'$ and w' is a permutation of $a_1 \dots a_l$, we have $\mathrm{supp}(\mathbf{x}') \cap \{a_1, \dots, a_\ell\}$ is nonempty. On the other hand, since $\pi_{w'}(\mathbf{x}'.\mathbf{q}') = \mathbf{x}.\mathbf{q}$, we have

$$\begin{aligned} \mathbf{x} =& \mathbf{x}' + \mathbf{M}_{w'}(\mathbf{q}') - |w'| = \mathbf{x}' + |a_{\ell+1}| - |a_1| \qquad \text{(by Lemma 8.6)} \\ =& \mathbf{x}'. \end{aligned}$$

In particular, we have $\mathrm{supp}(\mathbf{x}) = \mathrm{supp}(\mathbf{x}')$. Hence we conclude that $\mathrm{supp}(\mathbf{x}) \cap \{a_1, \dots, a_\ell\}$ is nonempty, as desired. $\square$

8.2 Counting recurrent components

In this section we turn to the problem of counting the number of recurrent components of an agent network.

We start with the following lemma. Recall the definition of capacity from Definition 5.14. Also recall that a configuration $\mathbf{x}.\mathbf{q}$ is stable if $\mathbf{x} \leq \mathbf{0}$, and is halting if there exists a stable configuration $\mathbf{x}'.\mathbf{q}'$ such that $\mathbf{x}.\mathbf{q} \longrightarrow \mathbf{x}'.\mathbf{q}'$.

Lemma 8.8. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected critical network.*

(i) *If $\mathcal{N}$ is an agent network, then* $\mathrm{cap}(\mathcal{N}) = 0$.

(ii) *If* $\mathrm{cap}(\mathcal{N}) = 0$ *and all states of $\mathcal{N}$ are locally recurrent, then $\mathcal{N}$ is an agent network.*

Proof. (i) By (8.1) the exchange rate vector $\mathbf{s}$ (Definition 5.13) of an agent network is equal to $\mathbf{1}$. By the definition of capacity, it suffices to show that any configuration $\mathbf{x}.\mathbf{q}$ of $\mathcal{N}$ with $\mathbf{1}^\top \mathbf{x} > 0$ does not halt.

Let $w \in A^*$ be any word and let $\mathbf{x}'.\mathbf{q}'$ be any configuration such that $\mathbf{x}.\mathbf{q} \xrightarrow{w} \mathbf{x}'.\mathbf{q}'$. Then

$$\mathbf{1}^\top \mathbf{x}' = \mathbf{1}^\top (\mathbf{x} + \mathbf{M}_w(\mathbf{q}) - |w|) = \mathbf{1}^\top \mathbf{x} + \mathbf{1}^\top \mathbf{M}_w(\mathbf{q}) - \mathbf{1}^\top |w| = \mathbf{1}^\top \mathbf{x} > 0,$$

where the third equality is due to $\mathcal{N}$ being an agent network. Hence $\mathbf{x}'.\mathbf{q}'$ is not a stable configuration. Since the choice of w and $\mathbf{x}'.\mathbf{q}'$ is arbitrary, this shows that $\mathbf{x}.\mathbf{q}$ does not halt, as desired.

(ii) Since $\mathrm{cap}(\mathcal{N}) = 0$, for any $a \in A$ and $\mathbf{q} \in Q$ the configuration $|a|.\mathbf{q}$ does not halt. In particular the letter a is not a complete execution for $|a|.\mathbf{q}$, and hence $\mathbf{1}^\top \mathbf{M}_a(\mathbf{q}) \geq 1$. Therefore, for all $w \in A^*$ and $\mathbf{q} \in Q$ we have $\mathbf{M}_w(\mathbf{q}) \geq \mathbf{1}^\top |w|$, and the equality is achieved only if $\mathbf{1}^\top \mathbf{M}_{w'}(\mathbf{q}) = \mathbf{1}^\top |w'|$ for all $w' \in A$ satisfying $|w'| \leq |w|$.

Let $\mathbf{r}$ be the period vector of $\mathcal{N}$. Note that for any $\mathbf{q} \in Q$,

$$\mathbf{1}^\top \mathbf{r} = \mathbf{1}^\top P \mathbf{r} = \mathbf{1}^\top \mathbf{M}_{\mathbf{r}}(\mathbf{q}) \geq \mathbf{1}^\top \mathbf{r},$$

where the second equality is due to the assumption that $\mathbf{q} \in \mathrm{Loc}(\mathcal{N}) = Q$, and the inequality is due to the conclusion in the previous paragraph. Since equality happens in the equation above and $\mathbf{r} \geq \mathbf{1}$, we conclude that $\mathbf{1}^\top \mathbf{M}_a(\mathbf{q}) = 1$ for all $a \in A$. Hence $\mathcal{N}$ is an agent network. $\square$

Remark. The condition in Lemma 8.8(ii) that every state in $\mathcal{N}$ is locally recurrent is necessary. Indeed, let $\mathcal{N}$ be a network with states $Q := \{\mathbf{q}_1, \mathbf{q}_2\}$, with alphabet $A := \{a\}$, and with transition functions given by

$$t_a(\mathbf{q}_1) = \mathbf{q}_2; \quad \mathbf{M}_a(\mathbf{q}_1) = 2|a|; \qquad t_a(\mathbf{q}_2) = \mathbf{q}_2; \quad \mathbf{M}_a(\mathbf{q}_2) = |a|.$$

This network has capacity zero, and yet is not an agent network since $\mathbf{1}^\top \mathbf{M}_a(\mathbf{q}_1) = 2$.

Recall that for any $m \in \mathbb{N}$, the set $\overline{\mathrm{Rec}}(\mathcal{N}, m)$ denotes the set of recurrent components (Definition 4.8) with level m. Also recall that $\mathrm{Tor}(\mathcal{N})$ denotes the torsion group of $\mathcal{N}$ (Definition 4.18).

Proposition 8.9. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected agent network. Then*

$$|\,\overline{\mathrm{Rec}}(\mathcal{N}, m)| = \begin{cases} 0 & \textit{if } m = 0; \\ |\,\mathrm{Tor}(\mathcal{N})| & \textit{if } m \geq 1. \end{cases}$$

Proof. By Lemma 5.4(ii) the level of a recurrent configuration is strictly positive, and by Lemma 5.19 the same is true for recurrent components. This proves the case when $m = 0$.

We now prove the case when $m \geq 1$. Since $\mathrm{cap}(\mathcal{N}) = 0$ by Lemma 8.8 and $\mathbf{s} = \mathbf{1}$ by equation (8.1), we have $\mathrm{Stop}(\mathcal{N}) = \{0\}$ by Lemma 5.23. Theorem 5.25(iii) then implies that $|\,\overline{\mathrm{Rec}}(\mathcal{N}, m)| = |\,\mathrm{Tor}(\mathcal{N})|$ for all $m \geq 1$, as desired. □

Proposition 8.9 can be compared to Theorem 7.11 in Chapter 7.2, which computes the same quantity for sandpile networks.

8.3 Determinantal generating functions for recurrent configurations

We now turn to the problem of counting the recurrent configurations of an agent network. We will derive two versions of a multivariate generating function identity.

The first identity counts recurrent configurations according to the number of chips at each vertex. For any $\mathbf{n} \in \mathbb{N}^A$ and $m \in \mathbb{N}$, we write

$$\mathrm{Rec}(\mathcal{N}, \mathbf{n}) := \{\mathbf{x}.\mathbf{q} \mid \mathbf{x}.\mathbf{q} \text{ is } \mathcal{N}\text{-recurrent and } \mathbf{x} = \mathbf{n}\}.$$

Let $(z_a)_{a \in A}$ be indeterminates indexed by A. We denote by $I(z)$ the $A \times A$ diagonal matrix with $I(z)(a,a) := \frac{1}{1-z_a}$ $(a \in A)$.

Theorem 8.10 (Determinantal formula for agent networks). *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected agent network. Then, in the ring of formal power series with $(z_a)_{a \in A}$ as indeterminates, we have the following identity:*

$$|\mathbb{Z}^A / K| \ \det\left(I(z) - P\right) = \sum_{\mathbf{n} \in \mathbb{N}^A} |\operatorname{Rec}(\mathcal{N}, \mathbf{n})| z^{\mathbf{n}}.$$

Theorem 8.10 can be compared to Theorem 7.4 in Chapter 7.1, which computes the same quantity for sandpile networks.

The second identity is a refinement of Theorem 8.10 for the special case of sinkless rotor networks, which involves edge variables that keep track of the rotor configuration.

For a digraph G, which may have multiple edges, let $(y_e)_{e \in E}$ and $(z_v)_{v \in V}$ be indeterminates indexed by edges of G and by vertices of G, respectively. We denote by $A_G(y)$ the weighted adjacency matrix indexed by V given by $A_G(y)(u, v) :=$

$\sum_e y_e$, where the sum is taken over all edges with source vertex v and target vertex u. We denote by $D_G(y,z)$ the diagonal matrix indexed by V with $D_G(y,z)(v,v) := \frac{1}{1-z_v} \sum_{e \in \mathrm{Out}(v)} y_e$. We denote by $\mathbb{Z}[y][[z]]$ the ring of formal power series in the $(z_v)_{v \in V}$ variables whose coefficients are polynomials in the $(y_e)_{e \in E}$ variables.

Theorem 8.11 (Master determinant for rotor networks). *Let $\mathcal{N}$ be a sinkless rotor network on a strongly connected digraph G. Then, in the ring $\mathbb{Z}[y][[z]]$ we have the following identity of formal power series:*

$$\det\left(D_G(y,z) - A_G(y)\right) = \sum_{\mathbf{x}.\mathbf{q} \in \mathrm{Rec}(\mathcal{N})} z^{\mathbf{x}}\, y_{\mathbf{q}},$$

where $y_{\mathbf{q}} := \prod_{v \in V} y_{\mathbf{q}(v)}$.

We remark that this identity is a refinement of the classical matrix tree theorem; see Appendix A.

We remark that Theorem 1.5 in §1.5 is a direct corollary of Theorem 8.11 by substituting $y_e = 1$ for all $e \in E$ and $z_v = z$ for all $v \in V$.

We now build towards the proof of these two theorems. We start with a lemma that refines Proposition 5.9 for agent networks.

Recall the definition of thief networks $\mathcal{N}_R$ from §5.2. Also recall the definition of recurrence for configurations (Definition 5.2) and states (Definition 4.26).

Lemma 8.12. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected agent network. Let $\mathbf{x} \in \mathbb{N}^A \setminus \{\mathbf{0}\}$ and let $R := A \setminus \mathrm{supp}(\mathbf{x})$. Then $\mathbf{x}.\mathbf{q}$ is an $\mathcal{N}$-recurrent configuration if and only if $\mathbf{q}$ is an $\mathcal{N}_R$-recurrent state.*

Proof. Let $\mathbf{r}$ be the period vector of $\mathcal{N}$. Note that $\mathrm{supp}((I - P_R)\mathbf{r}) = A \setminus R = \mathrm{supp}(\mathbf{x})$. By Corollary 8.5, the configuration $\mathbf{x}.\mathbf{q}$ is $\mathcal{N}$-recurrent if and only if $(I - P_R)\mathbf{r}.\mathbf{q}$ is $\mathcal{N}$-recurrent. The lemma now follows from Proposition 5.11. □

The following corollary of Lemma 8.12 generalizes the characterization of recurrent states for rotor networks with sinks in [43, Lemma 3.16].

Corollary 8.13. *Let $\mathcal{N}$ be a finite, locally irreducible, and strongly connected agent network, and let $R \subsetneq A$. Then $\mathbf{q} \in \mathrm{Loc}(\mathcal{N})$ is an $\mathcal{N}_R$-recurrent state if and only if every directed cycle in the rotor digraph $\varrho_{\mathbf{q}}$ contains a vertex in R.*

Proof. The corollary follows by applying Theorem 8.4 and Lemma 8.12 to the configuration $\mathbf{1}_R.\mathbf{q}$. □

We now quote a result from [17] that counts the number of recurrent states in a subcritical network.

Lemma 8.14 ([17, Theorem 3.3]). *Let $\mathcal{S}$ be a finite, locally irreducible, and subcritical abelian network with total kernel K and production matrix P. Then the number of recurrent states of $\mathcal{S}$ is equal to $|\mathbb{Z}^A/K|\det(I-P)$.* □

We now present the proof of Theorem 8.10. For an $A \times A$ matrix M and $R \subseteq A$, we denote by $\det(M;R)$ the determinant of the matrix obtained from deleting the rows and columns of M indexed by $A \setminus R$.

Proof of Theorem 8.10. Since $\mathrm{Rec}(\mathcal{N},\mathbf{0}) = \varnothing$ by Lemma 5.4(ii), we have

$$\sum_{\mathbf{n}\in\mathbb{N}^A} |\mathrm{Rec}(\mathcal{N},\mathbf{n})| z^{\mathbf{n}} = \sum_{R\subsetneq A} \sum_{\substack{\mathbf{n}\in\mathbb{N}^A;\\ \mathrm{supp}(\mathbf{n})=A\setminus R}} |\mathrm{Rec}(\mathcal{N},\mathbf{n})| z^{\mathbf{n}}.$$

Then

$$\sum_{\mathbf{n}\in\mathbb{N}^A} |\operatorname{Rec}(\mathcal{N},\mathbf{n})| z^{\mathbf{n}} = \sum_{R\subsetneq A} |\operatorname{Rec}(\mathcal{N}_R)| \prod_{a\in A\setminus R} \frac{z_a}{(1-z_a)} \qquad \text{(by Lemma 8.12)}$$

$$= \sum_{R\subsetneq A} |\mathbb{Z}^A/K| \det(I-P_R) \prod_{a\in A\setminus R} \frac{z_a}{1-z_a} \qquad \text{(by Lemma 8.14)}$$

$$= |\mathbb{Z}^A/K| \sum_{R\subsetneq A} \det(I-P;R) \det\left(I(z)-I; A\setminus R\right)$$

$$= |\mathbb{Z}^A/K| \det\left(I-P+I(z)-I\right) = |\mathbb{Z}^A/K| \det\left(I(z)-P\right). \qquad \square$$

We now build towards the proof of Theorem 8.11. A key ingredient in the refinement is the following extended version of the matrix tree theorem.

Let S be a subset of V. A subgraph $\mathcal{F}$ of G is a *directed forest rooted at* S if every vertex in S has outdegree 0, every vertex in $V\setminus S$ has outdegree 1, and the underlying graph of $\mathcal{F}$ has no cycles.

Lemma 8.15 (Extended matrix tree theorem [21]). *Let G be a digraph, and let S be a subset of V. Then*

$$\det(D_G(y,\mathbf{0}) - A_G(y); V\setminus S) = \sum_{\mathcal{F}} \prod_{e\in E(\mathcal{F})} y_e,$$

where the sum is taken over all directed forests of G rooted at S. $\square$

We now present the proof of Theorem 8.11.

Proof of Theorem 8.11. We have

$$\sum_{\mathbf{x}.\mathbf{q}\in\operatorname{Rec}(\mathcal{N})} z^{\mathbf{x}} y_{\mathbf{q}} = \sum_{S\subseteq A} \sum_{\substack{\mathbf{x}.\mathbf{q}\in\operatorname{Rec}(\mathcal{N});\\ \operatorname{supp}(\mathbf{x})=S}} z^{\mathbf{x}} y_{\mathbf{q}}.$$

Note that $\mathcal{N}$ is strongly connected since G is strongly connected. By Theorem 8.4, a configuration $\mathbf{x}.\mathbf{q}$ with $\operatorname{supp}(\mathbf{x}) = S$ is recurrent if and only if the digraph $\mathcal{F}$

given by

$$V(\mathcal{F}) = V(G), \qquad E(\mathcal{F}) = \{\mathbf{q}(v) \mid v \notin S\},$$

is a directed forest rooted at S. It then follows that

$$\begin{aligned}
&\sum_{\mathbf{x}.\mathbf{q}\in\mathrm{Rec}(\mathcal{N})} z^{\mathbf{x}}\, y_{\mathbf{q}} \\
&= \sum_{S\subseteq A} \det(D_G(y,\mathbf{0}) - A_G(y); V \setminus S) \prod_{v\in S} \sum_{e\in\mathrm{Out}(v)} \frac{y_e z_v}{1 - z_v} \quad \text{(by Lemma 8.15)} \\
&= \sum_{S\subseteq A} \det(D_G(y,\mathbf{0}) - A_G(y); V \setminus S) \det(D_G(y,z) - D_G(y); S) \\
&= \det(D_G(y,\mathbf{0}) - A_G(y) + D_G(y,z) - D_G(y)) \\
&= \det(D_G(y,z) - A_G(y)). \qquad \square
\end{aligned}$$

APPENDIX A

CLASSICAL MATRIX TREE THEOREM

In this chapter we give a short proof of the classical matrix tree theorem by using Theorem 8.4 and Theorem 8.11.

Let G be a strongly connected digraph. Recall that the Laplacian matrix Δ_G of G is the matrix $D_G - A_G$, where D_G is the outdegree matrix of G, and A_G is the adjacency matrix of G (§3.1). The *reduced Laplacian matrix* $\Delta_{G,s}$ of a vertex $s \in V$ is the matrix obtained by deleting the row and column of Δ that corresponds to s.

A *reverse arborescence* rooted at $s \in V$ is a directed subgraph T of G such that (1) T contains $|V(G)|-1$ edges, (2) $\text{outdeg}(s) = 0$ and $\text{outdeg}(v) = 1$ for any $v \in V(G) \setminus \{s\}$; and (3) for any vertex $v \in V(G) \setminus \{s\}$, there exists a directed path in T starting at v and ending at s.

Theorem A.1 (Classical matrix tree theorem [68]). *Let G be a strongly connected digraph, and let $s \in V$. Then the number of reverse arborescences of G rooted at s is* $\det(\Delta_{G,s})$.

Proof. Let $\mathcal{N}$ be the sinkless sandpile network on G. For any $\mathbf{q} \in Q$, we denote by $T_{\mathbf{q}}$ the directed subgraph obtained by deleting the outgoing edges of s from the rotor digraph $\rho_{\mathbf{q}}$ (Definition 8.1). By Theorem 8.4, we have the configuration $\mathbf{1}_s.\mathbf{q}$ is recurrent if and only the digraph $T_{\mathbf{q}}$ is a reverse arborescence of G rooted at s. Hence the number of reverse arborescenes of G rooted at s is equal to

$$\frac{1}{\text{outdeg}(s)} |\{\mathbf{q} \in Q \mid \mathbf{1}_s.\mathbf{q} \text{ is recurrent}\}|.$$

On the other hand, it follows from Theorem 8.11 that

$$|\{\mathbf{q} \in Q \mid \mathbf{1}_s.\mathbf{q} \text{ is recurrent}\}| = \frac{\partial}{\partial z_s}\bigg|_{z=\mathbf{0}} \det(D_G(z) - A_G) = \text{outdeg}(s)\det(\Delta_s).$$

This proves the theorem. $\square$

BIBLIOGRAPHY

[1] Arash Asadi and Spencer Backman. Chip-Firing and Riemann-Roch Theory for Directed Graphs. *ArXiv e-prints*, pages 1–29, December 2010.

[2] László Babai and Evelin Toumpakari. A structure theory of the sandpile monoid for directed graphs, 2010.

[3] Franco Bagnoli, Fabio Cecconi, Alessandro Flammini, and Alessandro Vespignani. Short-period attractors and non-ergodic behavior in the deterministic fixed-energy sandpile model. *EPL (Europhysics Letters)*, 63(4):512, 2003.

[4] Per Bak, Chao Tang, and Kurt Wiesenfeld. Self-organized criticality. *Phys. Rev. A (3)*, 38(1):364–374, 1988.

[5] Matthew Baker and Farbod Shokrieh. Chip-firing games, potential theory on graphs, and spanning trees. *J. Combin. Theory Ser. A*, 120(1):164–182, 2013.

[6] Georgia Benkart, Caroline Klivans, and Victor Reiner. Chip firing on dynkin diagrams and mckay quivers. *Mathematische Zeitschrift*, pages 1–34, Jan 2018.

[7] Abraham Berman and Robert J. Plemmons. *Nonnegative matrices in the mathematical sciences*. Academic Press [Harcourt Brace Jovanovich, Publishers], New York-London, 1979.

[8] N. L. Biggs. Chip-firing and the critical group of a graph. *J. Algebraic Combin.*, 9(1):25–45, 1999.

[9] Norman Biggs. Combinatorial Theory Seminar, *Mathematical Institute, Oxford University*, February 1997.

[10] Javier Bitar and Eric Goles. Parallel chip firing games on graphs. *Theoret. Comput. Sci.*, 92(2):291–300, 1992.

[11] Anders Björner, Bernhard Korte, and László Lovász. Homotopy properties of greedoids. *Adv. in Appl. Math.*, 6(4):447–494, 1985.

[12] Anders Björner and László Lovász. Chip-firing games on directed graphs. *J. Algebraic Combin.*, 1(4):305–328, 1992.

[13] Anders Björner, László Lovász, and Peter W. Shor. Chip-firing games on graphs. *European J. Combin.*, 12(4):283–291, 1991.

[14] Anders Björner and Günter M. Ziegler. Introduction to greedoids. In *Matroid applications*, volume 40 of *Encyclopedia Math. Appl.*, pages 284–357. Cambridge Univ. Press, Cambridge, 1992.

[15] Benjamin Bond and Lionel Levine. Abelian networks I. Foundations and examples. *SIAM J. Discrete Math.*, 30(2):856–874, 2016.

[16] Benjamin Bond and Lionel Levine. Abelian networks II: Halting on all inputs. *Selecta Math. (N.S.)*, 22(1):319–340, 2016.

[17] Benjamin Bond and Lionel Levine. Abelian networks III: The critical group. *J. Algebraic Combin.*, 43(3):635–663, 2016.

[18] Benjamin Braun, Hugo Corrales, Scott Corry, Luis David García Puente, Darren Glass, Nathan Kaplan, Jeremy L. Martin, Gregg Musiker, and Carlos E. Valencia. Counting arithmetical structures on paths and cycles. *ArXiv e-prints*, pages 1–22, Jan 2017.

[19] Robert Burton and Robin Pemantle. Local characteristics, entropy and limit theorems for spanning trees and domino tilings via transfer-impedances. *Ann. Probab.*, 21(3):1329–1371, 1993.

[20] Hannah Cairns. Some halting problems for abelian sandpiles are undecidable in dimension three. *ArXiv e-prints*, pages 1–28, August 2015.

[21] Seth Chaiken. A combinatorial proof of the all minors matrix tree theorem. *SIAM Journal on Algebraic Discrete Methods*, 3(3):319–329, 1982.

[22] Melody Chan, Thomas Church, and Joshua A. Grochow. Rotor-routing and spanning trees on planar graphs. *Int. Math. Res. Not. IMRN*, 2015(11):3225–3244, 2015.

[23] Swee Hong Chan. Abelian sandpile model and Biggs–Merino polynomial for directed graphs. *J. Combin. Theory Ser. A*, 154:145–171, 2018.

[24] Swee Hong Chan and Lionel Levine. Abelian networks IV. Dynamics of non-halting networks. *Memoirs of the American Mathematical Society*, to appear, 2019.

[25] Joshua Cooper, Benjamin Doerr, Joel Spencer, and Gábor Tardos. Deterministic random walks on the integers. *European J. Combin.*, 28(8):2072–2090, 2007.

[26] Robert Cori and Yvan Le Borgne. The sand-pile model and Tutte polynomials. *Adv. in Appl. Math.*, 30(1-2):44–52, 2003. Formal power series and algebraic combinatorics (Scottsdale, AZ, 2001).

[27] Hugo Corrales and Carlos E. Valencia. Arithmetical structures on graphs. *Linear Algebra and its Applications*, 536:120 – 151, 2018.

[28] Luca Dall'Asta. Exact solution of the one-dimensional deterministic fixed-energy sandpile. *Phys. Rev. Lett.*, 96:058003, Feb 2006.

[29] Arnaud Dartois and Dominique Rossin. Height arrow model. In *FPSAC PROCEEDINGS 2004 ACTES SFCA 2004 Vancouver CANADA*, page 87, 2004.

[30] Deepak Dhar. Self-organized critical state of sandpile automaton models. *Phys. Rev. Lett.*, 64(14):1613–1616, 1990.

[31] Deepak Dhar. Theoretical studies of self-organized criticality. *Phys. A*, 369(1):29–70, 2006.

[32] Leonard Eugene Dickson. Finiteness of the Odd Perfect and Primitive Abundant Numbers with n Distinct Prime Factors. *Amer. J. Math.*, 35(4):413–422, 1913.

[33] Benjamin Doerr and Mahmoud Fouz. Quasi-random rumor spreading: reducing randomness can be costly. *Inform. Process. Lett.*, 111(5):227–230, 2011.

[34] Matthew Farrell and Lionel Levine. CoEulerian graphs. *Proc. Amer. Math. Soc.*, 144(7):2847–2860, 2016.

[35] Anne Fey, Lionel Levine, and Yuval Peres. Growth rates and explosions in sandpiles. *J. Stat. Phys.*, 138(1-3):143–159, 2010.

[36] Robin Forman. Determinants of Laplacians on graphs. *Topology*, 32(1):35–46, 1993.

[37] C. M. Fortuin and P. W. Kasteleyn. On the random-cluster model. I. Introduction and relation to other models. *Physica*, 57:536–564, 1972.

[38] Andrei Gabrielov. Abelian avalanches and Tutte polynomials. *Phys. A*, 195(1-2):253–274, 1993.

[39] Eric Goles and Maurice Margenstern. Universality of the chip-firing game. *Theoret. Comput. Sci.*, 172(1-2):121–134, 1997.

[40] Pierre Antoine Grillet. *Commutative semigroups*, volume 2 of *Advances in Mathematics (Dordrecht)*. Kluwer Academic Publishers, Dordrecht, 2001.

[41] Pierre Antoine Grillet. Irreducible actions. *Period. Math. Hungar.*, 54(1):51–76, 2007.

[42] Johnny Guzmán and Caroline Klivans. Chip-firing and energy minimization on M-matrices. *J. Combin. Theory Ser. A*, 132:14–31, 2015.

[43] Alexander E. Holroyd, Lionel Levine, Karola Mészáros, Yuval Peres, James Propp, and David B. Wilson. Chip-firing and rotor-routing on directed graphs. In *In and out of equilibrium. 2*, volume 60 of *Progr. Probab.*, pages 331–364. Birkhäuser, Basel, 2008.

[44] Alexander E. Holroyd, Lionel Levine, and Peter Winkler. Abelian logic gates. *ArXiv e-prints*, pages 1–33, November 2015.

[45] Alexander E. Holroyd and James Propp. Rotor walks and Markov chains. In *Algorithmic probability and combinatorics*, volume 520 of *Contemp. Math.*, pages 105–126. Amer. Math. Soc., Providence, RI, 2010.

[46] Gérard Huet. Confluent reductions: abstract properties and applications to term rewriting systems. *J. Assoc. Comput. Mach.*, 27(4):797–821, 1980.

[47] Bálint Hujter, Viktor Kiss, and Lilla Tóthmérész. On the complexity of the chip-firing reachability problem. *Proc. Amer. Math. Soc.*, 145(8):3343–3356, 2017.

[48] Tian-Yi Jiang. On the Period Lengths of the Parallel Chip-Firing Game. *ArXiv e-prints*, pages 1–13, March 2010.

[49] Tian-Yi Jiang, Ziv Scully, and Yan X. Zhang. Motors and impossible firing patterns in the parallel chip-firing game. *SIAM J. Discrete Math.*, 29(1):615–630, 2015.

[50] Richard Kenyon. Spanning forests and the vector bundle Laplacian. *Ann. Probab.*, 39(5):1983–2017, 2011.

[51] Marcos A. Kiwi, René Ndoundam, Maurice Tchuente, and Eric Goles. No polynomial bound for the period of the parallel chip firing game on graphs. *Theoret. Comput. Sci.*, 136(2):527–532, 1994.

[52] Serge Lang. *Algebra*, volume 211 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, third edition, 2002.

[53] Lionel Levine. Parallel chip-firing on the complete graph: devil's staircase and Poincaré rotation number. *Ergodic Theory Dynam. Systems*, 31(3):891–910, 2011.

[54] Lionel Levine. Threshold state and a conjecture of Poghosyan, Poghosyan, Priezzhev and Ruelle. *Comm. Math. Phys.*, 335(2):1003–1017, 2015.

[55] Dino J. Lorenzini. Arithmetical graphs. *Math. Ann.*, 285(3):481–501, 1989.

[56] Russell Lyons and Yuval Peres. *Probability on trees and networks*, volume 42 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, New York, 2016.

[57] Criel Merino López. Chip firing and the Tutte polynomial. *Ann. Comb.*, 1(3):253–259, 1997.

[58] Cristopher Moore and Martin Nilsson. Parallel quantum computation and quantum codes. *SIAM J. Comput.*, 31(3):799–815, 2001/02.

[59] Kévin Perrot and Trung Van Pham. Feedback arc set problem and NP-hardness of minimum recurrent configuration problem of chip-firing game on directed graphs. *Ann. Comb.*, 19(2):373–396, 2015.

[60] Trung Van Pham. Orbits of rotor-router operation and stationary distribution of random walks on directed graphs. *Adv. in Appl. Math.*, 70:45–53, 2015.

[61] Alexander Postnikov and Boris Shapiro. Trees, parking functions, syzygies, and deformations of monomial ideals. *Trans. Amer. Math. Soc.*, 356(8):3109–3142 (electronic), 2004.

[62] Vyatcheslav B Priezzhev, Deepak Dhar, Abhishek Dhar, and Supriya Krishnamurthy. Eulerian walkers as a model of self-organized criticality. *Physical Review Letters*, 77(25):5079, 1996.

[63] James Propp. Random walk and random aggregation, derandomized. https://www.microsoft.com/en-us/research/video/random-walk-and-random-aggregation-derandomized/, 2003. Online Lecture.

[64] Eugene R. Speer. Asymmetric abelian sandpile models. *J. Statist. Phys.*, 71(1-2):61–74, 1993.

[65] Richard P. Stanley. *Combinatorics and commutative algebra*, volume 41 of *Progress in Mathematics*. Birkhäuser Boston, Inc., Boston, MA, second edition, 1996.

[66] Benjamin Steinberg. A theory of transformation monoids: combinatorics and representation theory. *Electron. J. Combin.*, 17(1):Research Paper 164, 56, 2010.

[67] Lilla Tóthmérész. Algorithmic aspects of rotor-routing and the notion of linear equivalence. *Discrete Applied Mathematics*, 236:428 – 437, 2018.

[68] W. T. Tutte. The dissection of equilateral triangles into equilateral triangles. *Proc. Cambridge Philos. Soc.*, 44:463–482, 1948.

[69] W. T. Tutte. Graph-polynomials. *Adv. in Appl. Math.*, 32(1-2):5–9, 2004. Special issue on the Tutte polynomial.

[70] Israel A. Wagner, Michael Lindenbaum, and Alfred M. Bruckstein. Smell as a computational resource—a lesson we can learn from the ant. In *Israel Symposium on Theory of Computing and Systems (Jerusalem, 1996)*, pages 219–230. IEEE Comput. Soc. Press, Los Alamitos, CA, 1996.